UNIVERSIDAD POLITÉCNICA DE MADRID

Facultad de Informática

Departamento de Matemática Aplicada

TESIS DOCTORAL

COMPLEJIDAD Y ESTRUCTURAS DE DATOS PARA EL PROBLEMA DE LOS RANGOS VARIABLES

Autora: Adriana Toni Delgado

Directora: Carmen Torres Blanc

Madrid, 2003

Agradecimientos

Bueno Juan...¡va por ti! No hay palabras para agradecer tanto apoyo.

Muchas gracias, Mamen, por lanzarte al ruedo y ayudarme tanto. También a vosotros, amigos: Antonio, Aglae, Emilio, Joaquín, Juancas, Sonia, Víctor y Victoria.

Compañeras Pili, Susana, Yolanda...pues eso.

También quiero mencionar a mis compañeros de departamento, que me han preguntado tantas veces *¡ a ver qué tal va todo!*, y a los compañeros de la secretaría, que se desviven por hacernos las cosas fáciles.

No a la Guerra

Capítulo 1

Introducción

1.1 Presentación del Problema

En esta tesis trabajamos sobre un problema computacional de formulación muy sencilla. La adición de distintos requisitos al enunciado general, abre un abanico de problemas distintos, de los cuales abordaremos un grupo variado y representativo.

El planteamiento general es el siguiente:

Se tiene un vector V de longitud n sobre el que deseamos ejecutar dos tipos de operaciones: de *acceso* y de *modificación*. En adelante nos referiremos a ellas con el nombre de operaciones *Retrieve* y operaciones *Update* respectivamente.

Las operaciones vienen definidas por

- $Retrieve(i,j):$ output $\sum_{z=i}^{j} v_z, \qquad 1 \leq i \leq j \leq n.$ (1)
- $Update(j,x):$ $v_j \longleftarrow v_j + x, \qquad j = 1 \ldots n.$

Se trata de diseñar estructuras de datos con sus programas correspondientes, para implementar las operaciones *Retrieve* y *Update*, así como de establecer cotas inferiores para la complejidad de las operaciones. Nos referimos a este problema con el nombre de *problema de los rangos variables* de tamaño n. [1]

[1] En inglés, habitualmente: "range queries"

Aclaramos un poco la motivación. Supongamos que redefinimos las operaciones como sigue:

- $Update(j, x)$: for $i = j$ to n do $[v_i \longleftarrow v_i + x]$, $j = 1 \ldots n$.
- $Retrieve(i, j)$: output $(v_j - v_{i-1})$, $1 \leq i \leq j \leq n$.

Si ejecutamos una secuencia $\sigma_1 \ldots \sigma_m$ de operaciones *Update* y *Retrieve* consecutivas sobre el vector inicialmente vacío, los outputs producidos - en las operaciones *Retrieve* - son los mismos, independientemente de que hayamos definido las operaciones de una u otra forma. Por tanto, ambas implementaciones resuelven el mismo problema computacional. Sin embargo, la primera de ellas implica una complejidad constante para las operaciones *Update* y lineal en el peor caso para las operaciones *Retrieve*, mientras que la segunda implica lo contrario.

De la misma manera, podemos utilizar estructuras de datos distintas al propio vector, por ejemplo árboles o grafos, de forma que una operación *Update* se ejecuta incrementando el valor almacenado en ciertos nodos, y una operación *Retrieve* sumando el valor almacenado en un subconjunto de ellos. De esta forma es posible obtener soluciones en las que el coste (en el peor caso o en promedio) de las operaciones de uno y otro tipo, está más equilibrado. Obviamente, independientemente de la estructura de datos que utilicemos, dar prioridad a la eficiencia de uno de los dos tipos de operación, va en detrimento de la eficiencia en la otra.

Como dijimos al principio, añadiendo condiciones al enunciado general del problema, obtenemos distintas *familias* de problemas. Vamos a establecer una primera distinción.

A lo largo de esta tesis, hablaremos en ocasiones de *el problema de los rangos variables*, y en otras de *el problema de las sumas parciales* [2]. Este último no es más que un caso particular del primero, en el que el primer argumento de la operación *Retrieve* - el extremo inferior del intervalo a sumar - es fijo, tomando

[2] En inglés: "partial sums problem"

siempre el valor 1. La definición de esta operación sería por tanto

$$Retrieve(k): \quad \text{output} \sum_{i=1}^{k} v_i, \qquad 1 \leq k \leq n.$$

Es más difícil obtener conclusiones para el problema de los rangos variables que para el problema de las sumas parciales, como es natural, y eso se corresponde con los resultados obtenidos hasta el momento para uno u otro problema.

Podemos establecer igualmente otra división del problema en dos familias distintas, tomando como criterio el *tipo de valores* que almacena el vector. De nuevo este factor tiene una gran incidencia en la dificultad de los problemas relacionados. Dado que las operaciones sobre el vector se definen exclusivamente a partir de la operación suma, las dos opciones son asumir que el vector almacena elementos de un *semigrupo conmutativo* o bien de un *grupo conmutativo* arbitrarios. Así, hablaremos de *el problema de los rangos variables en el caso grupo* o de *el problema de los rangos variables en el caso semigrupo*, y lo mismo para el *problema de las sumas parciales*. El problema es mucho más complicado en el *caso grupo*, en el que las implementaciones pueden utilizar la operación resta, y el número de problemas que permanecen abiertos es mucho mayor que en el *caso semigrupo*.

El estudio del problema de las sumas parciales y el problema de los rangos variables, se realiza en el marco de distintos modelos. Nosotros trabajaremos en el marco de modelos previamente definidos y también definiremos modelos propios equivalentes, que nos permiten obtener resultados nuevos de manera más sencilla.En ambos casos, consideramos modelos que incluyen las soluciones consistentes en un conjunto de variables $z_1 \ldots z_m$ (m puede variar) agrupadas en una cierta estructura de datos y tales que una operación *Retrieve* se ejecuta sumando un subconjunto de dichas variables , y una operación *Update* incrementando un subconjunto de ellas.

Como bibliografía básica relacionada con el estudio del problema dentro de este modelo de soluciones, citaremos [19], [15], [6] y [14].

Se conocen resultados para el problema de las sumas parciales dentro de un modelo más general que incluye al modelo algebraico definido por M.L.Fredman:

se asume la existencia de variables $z_1, z_2 \ldots$ que almacenan valores en el semigrupo, y que permiten hacer operaciones del tipo $z_i \leftarrow z_j + z_k$. En este sentido se pueden consultar [15] y [35].

El problema de los rangos variables es uno más de un amplio grupo catalogado como *problemas sobre estructuras de datos dinámicas.* Un problema de estructuras de datos dinámicas consiste en la representación en memoria de una colección finita de datos, de forma que que se puedan ejecutar cierto tipo de modificaciones de los datos (operaciones *update*) y cierto tipo de consultas sobre los datos (operaciones *retrieve*). Como ejemplos de otros problemas sobre estructuras de datos dinámicas de fácil formulación y amplias aplicaciones podemos citar:

- Representación de listas: se ocupa de la representación de una lista ordenada de como mucho n elementos pertenecientes al intervalo $1 \ldots n$, no necesariamente distintos, de forma que la operación de acceso $retrieve(k)$, devuelve el k-ésimo elemento de la lista, y la operación de modificación $inserta(k, u)$ inserta el elemento u entre los elementos situados en las posiciones $k-1$ y k.

- Rangos en un subconjunto: se trata de representar un subconjunto del conjunto $\{1 \ldots n\}$ permitiendo una operación de modificación que inserta un elemento en el conjunto, y una operación de acceso, $rango(j)$, que devuelve el número de elementos del subconjunto que son menores o iguales que j.

- Problema de la unión de conjuntos: se trata de diseñar una estructura de datos para manipular conjuntos de la siguiente manera. Inicialmente existen n conjuntos unitarios $\{1\}, \{2\} \ldots \{n\}$, siendo precisamente i el nombre del conjunto $\{i\}$. Se desea ejecutar una operación de acceso, $Encuentra(j)$, que devuelve el nombre del conjunto que contiene al elemento j, y una operación de modificación, $Union(A, B)$, combina los conjuntos de nombres A y B.

- *Orthogonal range queries:* se tiene un conjunto de registros cuyas claves son vectores de d dimensiones, pertenecientes a un espacio ordenado de

claves. Cada registro tiene asociado un valor. Se desea ejecutar operaciones de inserción y borrado, y como operación de acceso se desea una operación que devuelva la suma de los valores asociados a los registros en un hipercubo dado (producto cartesiano de intervalos). El problema varía según el tipo de elementos almacenados en los registros - pertenecientes a un grupo o semigrupo.

Una solución para un problema dinámico consiste en representar en memoria los datos y diseñar algoritmos correctos y a ser posible eficientes que implementen las operaciones.

Muchos de estos problemas han sido estudiados, y se estudian actualmente, dentro del llamado *cell probe model of computation.* En dicho modelo, la representación en memoria se modeliza como una secuencia de celdas $C_0, C_1, C_2 \ldots$ cada una de las cuales almacena un string de w bits. El parámetro w es el *tamaño de palabra* del modelo. Inicialmente todas las celdas contienen el string cero. La complejidad en tiempo de ejecución de un cómputo secuencial se define como el número de palabras de memoria accedidas. Existe muchísima bibliografía que trata de distintos problemas abordados dentro de este modelo, entre ellos el problema del que nos hemos ocupado en esta tesis y los mencionados más arriba.Como pequeña muestra citaremos [18], [3], [8] , [9], [11], [16], [26], [28].

Este modelo es aplicable también a problemas sobre estructuras de datos estáticas. Los ejemplos más naturales de problemas sobre estructuras de datos estáticas son los problemas de búsqueda, como por ejemplo:

- Pertenencia : dado el universo $U = \{1 \ldots m\}$ y un subconjunto $S \subseteq U$ con $|S| = n$, se desea almacenar S como una estructura de datos e implementar eficientemente la operación que decide la pertenencia o no de un elemento de U al conjunto S.

- Diccionario: dado el universo $U = \{1 \ldots m\}$ y un subconjunto $S \subseteq U$, asociamos a cada elemento $x \in S$ una información $v_x \in U$. Se desea almacenar S y la información asociada como una estructura de datos que permita la ejecución eficiente de una operación que decide si un elemento dado está o no en S, y en caso de ser así devuelva el valor asociado v_x.

Cuando se trabaja con estructuras de datos estáticas, el tamaño de la estructura de datos se convierte en un parámetro crucial. Si no hay restricciones en cuanto al espacio de almacenamiento, se tiene siempre, para cualquier problema, una solución con tiempo de ejecución constante para las operaciones de consulta de la estructura: ¡simplemente hay que tener una tabla en la que buscar la respuesta a cada posible pregunta! En realidad, en estos problemas el interés se centra en estudiar la relación entre el espacio de almacenamiento y el tiempo de ejecución. En relación a este tema se pueden consultar entre otros [21], [17], [25], [5], [10], [34], [36].

1.2 Resultados Obtenidos

Los resultados obtenidos en esta tesis se organizan en cuatro capítulos que pasamos a describir brevemente. Antes, mencionar que en el Capítulo 2 se expone un repaso de conceptos relacionados con la complejidad de algoritmos - cota inferior, complejidad en media, peor caso, cota inferior optimal etc - que incluye todos los mencionados a continuación, y también la descripción del *modelo algebraico* para el estudio del *problema de las sumas parciales* que aquí se cita.

- Capítulo 3: trabajamos dentro del *modelo algebraico* definido por M.L.Fredman (ver [14]) para el estudio del *problema de las sumas parciales*, dentro del *caso semigrupo*. Presentamos un método matricial para la obtención de cotas inferiores para la complejidad media de las operaciones, que permite alcanzar la cota optimal de $\log_4 n + \frac{\log_4 n}{n} + \frac{1}{n}$ para un tamaño n del problema. Existen estructuras de datos optimales previamente definidas, que ofrecen exactamente esa complejidad promedio, y por eso afirmamos que nuestro método permite alcanzar la mejor cota posible.

- Capítulo 4: presentamos un nuevo modelo, al que llamamos *modelo orbital*, para definir soluciones al *problema de las sumas parciales* dentro del *caso semigrupo*. El modelo requiere menos espacio en memoria para la descripción de las soluciones que el *modelo algebraico* mencionado en el

punto anterior, en particular $\Theta(n)$ frente a $\Theta(n^2)$. Existen soluciones dentro del *modelo algebraico* que no se pueden acomodar en el *modelo orbital*, pero probaremos que nuestro modelo incluye soluciones optimales en términos de la complejidad media de las operaciones. En la parte final del capítulo, se adapta el modelo de forma que acomode soluciones al *problema de los rangos variables*. También se define una clase de soluciones para este problema, dentro del *modelo orbital* adaptado, que ofrece una complejidad media de $\Theta(logn)$.

- Capítulo 5: es el capítulo más voluminoso de esta tesis, y el que presenta un mayor número de resultados nuevos. Nos ocupamos ahora del *problema de los rangos variables* dentro del *caso semigrupo*. Definimos dos nuevos modelos en los que enmarcar el estudio del problema, a los que llamaremos *modelo matricial* y *modelo de grafos*. Dichos modelos nos permiten definir soluciones que disminuyen la complejidad media que ofrecen las estructuras de datos definidas hasta el momento, así como el número de variables utilizadas. Las soluciones se definen a través de un par de matrices en el *modelo matricial*, y mediante un grafo en el *modelo de grafos*, en ambos casos junto con los programas correspondientes para la implementación de las operaciones *Update* y *Retrieve*. En el proceso de formalización de ambos modelos se desarrollan numerosos algoritmos que consideramos interesantes. Además, trabajando dentro de nuestro *modelo matricial*, desarrollamos un sencillo procedimiento para calcular una cota inferior para la suma de la complejidad media de las operaciones *Retrieve* y la complejidad media de las operaciones *Update* - por separado - que mejora ligeramente la cota establecida a este respecto hasta este momento.

- Capítulo 6: reformulamos un problema que permanece abierto dentro del *problema de las sumas parciales* en el *caso grupo*. Se trata de establecer una cota inferior optimal para la complejidad media de las operaciones. Trabajamos en el marco del *modelo algebraico* de M.L.Fredman ya mencionado ([14]). La reformulación del problema plantea un problema distinto y difícil igualmente, pero pensamos que puede sugerir nuevas vías de ataque.

En la parte inicial de cada uno de estos capítulos, se introducen la notación y los conceptos específicos para el desarrollo del mismo, y se exponen los resultados ya conocidos que tengan relación con los que nosotros vamos a presentar.

Los conceptos y la notación generales, de aplicación en distintos capítulos, se exponen en el Capítulo 2.

Capítulo 2

Conceptos Básicos

En este capítulo presentamos conceptos necesarios para la lectura y comprensión de esta tesis. En la mayoría de los casos, se trata simplemente de recordar conceptos básicos ampliamente conocidos, del campo de las matemáticas, las estructuras de datos y la complejidad de algoritmos.

Presentamos también conceptos específicos relacionados con el problema que nos ocupa, pero sólo aquellos que afectan al conjunto de la tesis, pues los que son de aplicación tan sólo en algún capítulo concreto, se introducirán en dicho capítulo.

Organizamos el capítulo en tres apartados distintos:

- Conceptos relacionados con la complejidad de algoritmos.
- Conceptos básicos de álgebra lineal.
- Conceptos relativos al problema computacional que estamos tratando.

2.1 Conceptos Relacionados con la Complejidad de Algoritmos

Recordamos en este apartado el significado de las distintas notaciones que se utilizan para tratar con funciones que expresan complejidad de algoritmos, y los criterios que rigen el estudio de los distintos tipos de complejidad (media, peor caso...) o de cotas para la complejidad.

Como bibliografía de consulta sobre cuestiones relacionadas con el análisis de la complejidad de algoritmos, se puede consultar [22], [4] y [7].

- **Complejidad en el peor caso:** el análisis de la complejidad en el peor caso define el coste de aplicación de un algoritmo a un *problema de tamaño* n como el máximo coste sobre todos los problemas de tamaño n. Como consecuencia, si f es una función de complejidad basada en un análisis del peor caso, entonces para cada problema de tamaño n, el coste de aplicación del algoritmo es menor o igual que $f(n)$.

- **Complejidad media:** el análisis del caso medio, cuando todos los problemas de tamaño n son equiprobables, define el coste de aplicación de un algoritmo a un problema de tamaño n, es decir $f(n)$, como la suma de los costes de aplicación del algoritmo a cada uno de los problemas de tamaño n dividido por el número de problemas de esa talla.

- **Cota inferior para la complejidad:** hablamos de cota inferior para la complejidad *de un problema*, no de un algoritmo concreto que lo resuelva. Se trata de establecer una cota inferior para la complejidad de cualquier posible algoritmo solución del problema. Si la función f es una cota inferior para la complejidad (*caso peor* o *caso medio*) de un problema P, entonces para cada *tamaño del problema* n, y para cada algoritmo A que resuelva el problema P, se tiene que el coste de aplicación de A (*caso peor* o *caso medio*) para el problema de tamaño n, es mayor o igual que $f(n)$.

- **Cota superior para la complejidad:** el concepto de cota superior en complejidad va ligado a un algoritmo concreto. Se trata simplemente de establecer una función que acote superiormente a la función de compleji-

dad f asociada al algoritmo. De nuevo hay que considerar que f puede estar basada en un análisis del *peor caso* o el *caso medio.*

- **Cota inferior optimal:** decimos que una cota inferior para la complejidad de un problema (*caso peor* o *caso medio*) es *optimal* si existe un algoritmo que resuelve dicho problema cuya función de complejidad (*caso peor* o *caso medio*) es precisamente dicha cota inferior.

- **Notación O()** : se pueden ignorar los factores constantes cuando se trata de evaluar la complejidad de un determinado algoritmo. La notación *O()* nos permite hacerlo. Decimos que una función $g(n)$ es $O(f(n))$ - siendo $f(n)$ otra función - si existen constantes c y N tales que para todo $n \geq N$, se tiene $g(n) \leq cf(n)$. En otras palabras, para n lo suficientemente grande, el valor de g es menor o igual que una cierta constante por el valor de f.

  La notación $O()$ nos permite ignorar las constantes convenientemente. Podemos incluir constantes dentro de esta notación, pero no tiene sentido hacerlo. Siempre escribiremos $O(n)$ en lugar de $O(5n+3)$, por ejemplo. Recordamos que el conjunto $O(\log n)$ es independiente de la base del logaritmo, que por eso no aparece expresada cuando utilizamos esta notación.

  Escribimos $O(1)$ para denotar una constante.

  La notación $O()$ sólo acota por arriba. Por ejemplo, si $f(n) = 5n^2 + 12$, podemos decir que f pertenece al conjunto $O(n^2)$, pero también podemos decir que pertenece a $O(n^3)$, $O(2^n)$ etc, en general podemos decir que pertenece a $O(h(n))$ para cualquier función h que verifique que $\lim_{n\to\infty} \frac{f(n)}{h(n)}$ es una constante, incluído el valor cero. La notación $\Theta()$, de la que hablaremos dos puntos más adelante, nos permitirá una mayor precisión.

- **Notación Ω()** : de la misma manera que la notación $O()$ nos da una cota superior asintótica para una función, la notación $\Omega()$ nos da una cota inferior asintótica. Decimos que una función $g(n)$ es $\Omega(f(n))$ si existen

constantes c y N tales que para todo $n \geq N$, se tiene $g(n) \geq cf(n)$. En otras palabras, para n lo suficientemente grande, el valor de g es mayor que una cierta constante por el valor de f.

- **Notación $\Theta()$** : podemos definir el significado de esta notación en función de los dos tipos de notación que acabamos de describir. Así, decimos que $g(n)$ pertenece al conjunto de funciones $\Theta(f(n))$ si y sólo si $g(n)$ pertenece a $O(f(n))$ y también a $\Omega(f(n))$. Es decir, la notación $\Theta()$ acota asintóticamente una función por arriba y por abajo.

2.2 Conceptos de Álgebra Lineal

En este apartado recordamos conceptos elementales sobre aplicaciones lineales y matrices, a los que se hace referencia en la tesis. Se aplican fundamentalmente en los Capítulos 3 y 6.

Como bibliografía de consulta sobre los temas de álgebra relacionados con este trabajo, podemos recomendar especialmente [23], [24] y [27]. La primera de las tres referencias es útil para consultar cuestiones relativas a estructuras algebraicas, más en concreto teoría de grupos, y en todo caso sería de utilidad exclusivamente para el Capítulo 6, en el que trabajamos en $\mathbf{Z_2}$ (los enteros módulo 2). Las nociones de teoría de grupos necesarias para la lectura de ese capítulo son muy elementales, y por eso en este apartado hemos incluído sólo referencias al álgebra lineal.

- **Semigrupo commutativo:** conjunto de elementos cerrado bajo una operación binaria commutativa y asociativa para la cual existe un elemento neutro en el conjunto .

- **Grupo commutativo:** conjunto de elementos cerrado bajo una operación binaria commutativa y asociativa para la cual existe un elemento neutro en el conjunto y tal que todo elemento tiene inverso.

- **Rango de una aplicación lineal:** el rango de una aplicación lineal entre dos espacios vectoriales (el primero de ellos de dimensión finita), es la dimensión de la imagen.

  El rango es igual a la dimensión del espacio origen menos la dimensión del *kernel* de la aplicación ($\ker f = f^{-1}(0)$).

  Una aplicación lineal entre espacios finitos de la misma dimensión es biyectiva si y solo si el rango es igual a la dimensión del espacio vectorial origen (finito).

- **Correspondencia entre una aplicación lineal y una matriz:** sean dos espacios vectoriales E, F de dimensiones n y m, y sean $e_1 \dots e_n$ y $f_1 \dots f_m$ bases respectivas para dichos espacios. Existe una *biyección* entre las aplicaciones lineales de E en F y las matrices de dimensión $m \times n$.

  A cada aplicación lineal f le corresponde la matriz $A = (\alpha_{i,j})$ de dimensión $m \times n$, siendo $f(e_k) = \sum_{i=1}^{m} \alpha_{i,k} f_i$ para todo $k = 1 \dots n$.

  De la misma manera, a partir de una matriz $A = (\alpha_{i,j})$ y fijada una base $e_1 \dots e_n$ para E, se define la aplicación lineal f tal que si $x = \sum_{k=1}^{n} \xi_k e_k$ entonces $f(x) = \sum_{i=1}^{m} \eta_i f_i$ con $\eta_i = \sum_{k=1}^{n} \alpha_{i,k} \xi_k$ para todo $i = 1 \dots m$.

  Si a la aplicación lineal f le corresponde la matriz A, entonces el *rango* de f es igual al *rango* por columnas de A - máximo número de columnas linealmente independientes. Podemos hablar simplemente del *rango* de f, pues el rango por filas y columnas es el mismo.

  Si E y F tienen la misma dimensión, entonces f es biyectiva si y sólo si el determinante de A es distinto de cero.

- **Matriz del cambio de base en un espacio vectorial:** sea E un espacio vectorial de dimensión n, y sean $e_1 \dots e_n$ y $f_1 \dots f_n$ dos bases para E.

  Denotamos con $\xi_1 \dots \xi_n$ a las coordenadas de un cierto vector $x \in E$ respecto a la primera base, y con $\eta_1 \dots \eta_n$ a las coordenadas respecto de

la segunda. Naturalmente, las dos bases están relacionadas de la siguiente forma:

$$f_i = \sum_{k=1}^{n} \sigma_{i,k} e_k \qquad i = 1 \dots n,$$

es decir,

$$f = Se \qquad \text{con} \qquad S = (\sigma_{i,k})$$

siendo e y f las matrices cuyas columnas son los vectores $e_1 \dots e_n$ y $f_1 \dots f_n$ respectivamente.

Llamamos a S *matriz del cambio de base,* y se tiene que $x = \xi' e = (\eta' S) e$, donde ξ', η' son respectivamente los vectores $(\xi_1 \dots \xi_n)$ y $(\eta_1 \dots \eta_n)$.

- **Cambios de base y aplicaciones lineales:** sea f una aplicación lineal de un espacio vectorial E de dimensión n en un espacio vectorial F de dimensión m, de forma que respecto a ciertas bases de E y F, la matriz correspondiente a f es $A = (\alpha_{i,j})$, de dimensión $m \times n$.

  Se tiene que $\eta = A\xi$, con $x = \sum_{k=1}^{n} \xi_k e_k \Rightarrow$ $f(x) = \sum_{k=1}^{n} \xi_k f(e_k) = \sum_{i=1}^{m} \left[\sum_{k=1}^{n} \alpha_{i,k} \xi_k\right] f_i = \sum_{i=1}^{m} \eta_i f_i$.

  Supongamos que se introducen nuevas bases para E y F, y supongamos que las nuevas componentes vectoriales, que denotamos por ξ^* en E y η^* en F, están relacionadas con las anteriores por las matrices de cambio de base S y T, de forma que $\xi = S\xi^*$, $\eta = T\eta^*$.

  Entonces tenemos que

  $$T\eta^* = AS\xi^*$$

  y multiplicando por la izquierda por T^{-1} (una matriz $n \times n$ corresponde a un cambio de base si y sólo si es inversible),

  $$\eta^* = (T^{-1}AS)\xi^*$$

  La aplicación f está representada por la matriz

  $$A^* = T^{-1}AS$$

  respecto a las nuevas bases.

En particular, si $E = F$, entonces $S = T$ y tenemos

$$A^* = S^{-1}AS$$

- **Endomorfismo:** un *endomorfismo* es una aplicación lineal de un espacio vectorial en si mismo.

- **Espacio vectorial *f-cíclico*. Vector *f-generador*:** un espacio vectorial E dotado de un endomorfismo f es *f-cíclico* si existe un vector $a \in E$ tal que $E = L(a, f(a), f^2(a)\dots)$, es decir, si todo vector de E se puede expresar como combinación lineal de vectores $a, f(a), f^2(a)\dots$

  En tal caso, decimos que a es un *f-generador* de E.

- **Subespacio vectorial *f-invariante*:** un subespacio vectorial L de E, es *f-invariante* si $f(L)$ está contenido en L.

- **Suma directa:** si L_1 y L_2 son subespacios vectoriales de E tales que $L_1 \cap L_2 = \{0\}$, llamamos *suma directa de* L_1 y L_2 a $L(L_1 \cup L_2)$, y la denotamos por $L_1 \oplus L_2$.

- **Subespacio vectorial *f-irreducible*:** un subespacio vectorial L de E, es *f-irreducible* si es *f-invariante* y no se puede descomponer en suma directa de subespacios *f-invariantes* distintos de $\{0\}$.

- **Polinomio característico:** dado un endomorfismo sobre un espacio vectorial E al que corresponde la matriz A, el *polinomio característico* es el determinante de $(A - \lambda I)$. El polinomio característico es independiente de la base del espacio vectorial.

  Las raíces del polinomio característico reciben el nombre de *autovalores.*

- **Polinomio minimal:** sea E un espacio vectorial de dimensión n sobre un cuerpo K, y sea f un endomorfismo de E. El polinomio $\mathbf{f}_f = \sum_{k=0}^{n} \alpha_k f^k$ es de nuevo un endomorfismo de E (f^0 es el endomorfismo identidad).

El polinomio $\mathbf{f} = 1$ es el endomorfismo identidad, y el polinomio $\mathbf{f} = 0$ es el endomorfismo cero.

Un polinomio es *mónico* si el coeficiente del término de mayor grado es 1.

El *polinomio minimal* de f es el único polinomio mónico $\mathbf{m}$ de menor grado, tal que $\mathbf{m}_f = 0$.

- **Polinomio *f-anulador*:** el *f-anulador* de un vector $x \in E$, es el único polinomio mónico $\mathbf{h}$ de menor grado, tal que $\mathbf{h}_f(x) = 0$.

2.3 Conceptos Relacionados con Nuestro Problema Computacional

En esta sección enunciamos formalmente el *problema de las sumas parciales*, en el caso grupo y semigrupo, y el *problema de los rangos variables* en el caso semigrupo. Para el primero de ellos, describimos también el *modelo algebraico* definido por M.L.Fredman (ver [14]), pues hablaremos del mismo en distintos capítulos.

2.3.1 El Problema de las Sumas Parciales

El *problema de las sumas parciales* trata del diseño de estructuras de datos y algoritmos para el mantenimiento de un vector sobre el que se desea realizar dos tipos de operaciones: operaciones de acceso y operaciones de modificación, a las que nos referimos respectivamente con el nombre de operaciones *Retrieve* y *Update*.

Se considera que el vector almacena valores bien de un semigrupo conmutativo (*caso semigrupo*) o bien de un grupo conmutativo (*caso grupo*) arbitrarios.

Llamando V al vector que se desea mantener y n a su longitud, definimos las operaciónes de la siguiente manera:

(a) *Update(i, x):* $v[i] \leftarrow v_i + x\ , \quad 1 \leq i \leq n,\ x \in G$

(b) *Retrieve(j):* output $\sum_{k=1}^{j} v_k \quad 1 \leq j \leq n$ (1)

Si se trabaja dentro del *caso semigrupo*, las implementaciones no pueden utilizar la operación resta.

Una posible interpretación de las operaciones podría ser la siguiente: se trata de almacenar las puntuaciones obtenidas por un gran número de individuos en una cierta prueba, a lo largo de un período de tiempo indefinido. Las puntuaciones posibles son los enteros comprendidos entre 1 y n. Cada vez que un nuevo individuo obtiene la puntuación j, se actualiza la información almacenada ejecutando la operación $Update(j, 1)$. La intención es que v_j represente el número total de individuos que han obtenido la puntuación j. La operación $Retrieve(k)$ permite conocer la cantidad de individuos que han obtenido una puntuación igual o menor que k.

Ahora supóngase que las operaciones se redefinen de la siguiente manera:

(a) *Update(j, x):* for $i := j$ until n do $[\ s_i := s_i + x\ ]$

(b) *Retrieve(k):* output s_k (2)

Si los vectores V y S son inicialmente el vector cero y se ejecuta una cierta secuencia $\sigma_1 \ldots \sigma_s$ de operaciones *Update* y *Retrieve*, se generarán los mismos output independientemente de que los programas se hayan definido de cualquiera de las dos maneras. Decimos, por tanto, que los programas definidos en (1) y (2) resuelven el mismo problema computacional. Sin embargo, en (1) una operación *Update* tiene complejidad constante, y una operación *Retrieve* tiene complejidad lineal, mientras que en (2) ocurre lo contrario.

Una tercera solución a este problema podría ser utilizar como estructura de datos un árbol binario con n hojas, en lugar de un vector. En la hoja j - de izquierda a derecha - almacenamos el número de individuos que han obtenido la puntuación j, y en cada nodo interno la suma de los valores en las hojas del subárbol que tiene como raíz el propio nodo. La operación $Update(j, x)$ se ejecuta añadiendo x a los valores de los nodos que van desde la raíz del árbol hasta la hoja j. La operación $Retrieve(k)$ se ejecuta sumando los valores

almacenados en el conjunto de nodos apropiado. En el peor caso, el número de nodos a los que es necesario acceder para ejecutar cualquier operación, es aproximadamente $\log_2 n$.

Estas tres soluciones parecen sugerir que existe una cierta relación entre las complejidades de ejecutar operaciones *Update* y *Retrieve*.

Modelo Algebraico

Describimos a continuación el *modelo algebraico* propuesto por M.L.Fredman (ver [14]) para el estudio de cotas inferiores y superiores para la complejidad de las operaciones.

El modelo incluye todas las soluciones en las que una operación de acceso *Retrieve*(j) se efectúa sumando un subconjunto de las variables que componen la estructura de datos solución $(z_1 \dots z_m)$, y una operación de modificación *Update* incrementando un subconjunto de dichas variables.

Consiste en triplas $< R, U, Z >$ donde $Z = (z_1 \dots z_m)$ es un conjunto de variables para el almacenamiento de valores del grupo o semigrupo conmutativo (dependiendo del *caso* en que nos encontremos) correspondiente, y $R = (r_{i,j})_{n \times m}$, $U = (u_{i,j})_{m \times n}$ son matrices de números enteros cuyas filas y columnas sirven respectivamente para describir las operaciones de acceso y modificación. Las matrices deben verificar $R \times U = T$ donde

$$T = (t_{i,j})_{i,j=1\dots n}, \quad \text{siendo} \quad t_{i,j} = \begin{cases} 1 & i \geq j \\ 0 & i < j \end{cases}$$

Los programas asociados a una tripla son

(a) *Update(j, x):* for $l := 1$ until m do $[\ z_l := z_l + u_{l,j}x\ ]$

(b) *Retrieve(k):* output $\sum_{l=1}^{m} r_{k,l} z_l$ (3)

Si se asume que se quieren almacenar valores de un grupo conmutativo, las matrices serán matrices de números enteros (positivos, negativos o cero), mientras que si nos ocupamos del caso semigrupo, las matrices serán de ceros y unos.

La pareja de matrices correspondiente - tanto en el caso grupo como en el caso semigrupo - a la definicion de las operaciones *Update* y *Retrieve* dada en (1) en esta misma sección, sería $R = T,\ U = I,$ mientras que la correspondiente a la definición de operaciones dada en (2) sería $R = I,\ U = T.$

Para el estudio de cotas inferiores para la complejidad de las operaciones es suficiente con considerar parejas de matrices sobre los enteros módulo 2 en el caso grupo, pues las cotas inferiores obtenidas para esa clase de factorizaciones son obviamente válidas en el caso general.

Incluímos aquí la demostración de que los programas definidos en (3) son correctos si y sólo si el producto de las matrices R y U es la matriz T, pues nos servirá como modelo para demostraciones similares en varios momentos a lo largo de la tesis.

El resultado aparece enunciado como *Lema 1* en [14] , y para demostrarlo, se considera la ejecución de la secuencia de operaciones $Retrieve(k),\ Update(j,x),\ Retrieve(k).$

Se denota como $w_1,\ w_2$ a los output

$$w_1 = \sum_{l=1}^{m} r_{k,l} z_l \qquad w_2 = \sum_{l=1}^{m} r_{k,l}(z_l + u_{l,j}x)$$

Es evidente que los programas definidos en (3) son correctos si y sólo si

$$w_2 - w_1 = \begin{cases} x & j \leq k \\ 0 & j > k \end{cases}$$

pero $w_2 - w_1 = (\sum_{l=1}^{m} r_{k,l} u_{l,j})\, x,$ y el lema se deduce inmediatamente.

2.3.2 El Problema de los Rangos Variables

Enunciamos ahora el *problema de los rangos variables* dentro del *caso semigrupo*, tal cual aparece en [19].

Se tiene un vector $V = (v_1, v_2, \ldots v_n)$ que almacena valores pertenecientes a un semigrupo conmutativo cualquiera S. Se pretende definir estructuras de datos y algoritmos que implementen operaciones de acceso y modificación sobre

el vector. Nos referiremos a tales operaciones con el nombre de operaciones *Retrieve* y *Update* respectivamente.

Las operaciones vienen definidas por :

(a) $Retrieve(j,k)$: output $\sum_{i=j}^{k} v_i \quad \forall\ 1 \leq j \leq k \leq n.$

(b) $Update(j,x)$: $v_j := v_j + x \quad \forall\ 1 \leq j \leq n, \quad x \in S$ (1)

Dado que el vector almacena valores pertenecientes a un semigrupo, las implementaciones no podrán utilizar la operación de la resta.

Según la interpretación de las operaciones sugerida en la sección 2.3.1 de este mismo capítulo, según la cual se trata de almacenar y consultar las puntuaciones obtenidas en una cierta prueba por un gran número de individuos a lo largo del tiempo, en el caso del problema de los rangos variables la ejecución de una operación $Retrieve(i,j)$ devolverá el número de individuos que han obtenido una puntuación entre i y j.

Se consideran estructuras de datos que involucran una colección finita de variables $\{z_1, z_2 \ldots z_m\}$ que toman valores del semigrupo S, de forma que cada variable z_i almacena el valor $\sum_{j \in Y_i} v_j$ donde Y_i es un subconjunto de $\{1, 2 \ldots n\}$.

La operación $Retrieve(j,k)$ se implementa con el programa

$$output \sum_{i \in R_{jk}} z_i$$

donde R_{jk} es un subconjunto de $\{1 \ldots m\}$.

La operación $Update(j,x)$ se implementa con el programa

$$z_l \leftarrow z_l + x \qquad \forall\ l \in U_j$$

siendo $U_j = \{i | j \in Y_i\}$.

En [19] se demuestra que una estructura de datos de esta clase es válida si y sólo si

$$|U_l \cap R_{jk}| = \begin{cases} 1 & \text{si } j \leq l \leq k \\ 0 & \text{e.o.c} \end{cases}$$

Dentro de este modelo, se define la complejidad de una operación $Update(j, x)$ como $|U_j|$, y la complejidad de $Retrieve(j, k)$ por $|R_{jk}|$.

El número distinto de operaciones $Retrieve(i, j)$ en función de las distintas parejas de argumentos posibles es $\frac{n(n+1)}{2}$. En cuanto a las operaciones $Update(j, x)$, el número de operaciones distintas a la hora de considerar la complejidad de la operación es n, pues la complejidad no depende del segundo argumento y el primero de ellos puede tomar valores entre 1 y n.

Capítulo 3

Método Matricial para la Obtención de una Cota Inferior para el Problema de las Sumas Parciales

En este capítulo nos ocupamos del *problema de las sumas parciales* dentro del modelo *semigrupo*. Presentamos un método algebraico para la obtención de una cota inferior para la complejidad media conjunta de las operaciones *Update* y *Retrieve*.

Trabajamos dentro del *modelo algebraico* definido por M.L. Fredman (ver [14] o el apartado 2.3.1 del Capítulo 2). Recordamos que consiste en utilizar parejas de matrices de ceros y unos R, U tales que su producto es la matriz triangular inferior T, para definir las operaciones *Retrieve* y *Update* respectivamente. En particular, la fila $i-esima$ de la matriz R define el conjunto de variables que se suman al ejecutar $Retrieve(i)$, y la columna $j-esima$ de la matriz U determina el conjunto de variables que se actualizan al ejecutar una operación $Update(j, \_)$.

Nuestro método nos permite establecer con absoluta precisión - incluyendo

factores constantes - la cota inferior optimal para la complejidad media de las operaciones, es decir, la mayor cota inferior posible. Lo que hacemos es determinar el número mínimo de unos que debe tener cualquier pareja de matrices R, U de ceros y unos cuyo producto sea la matriz T.

En realidad, es posible deducir la cota *a partir de resultados obtenidos anteriormente.* En efecto, en [15] se definen árboles binarios solución para el *problema de las sumas parciales* y se prueba que son optimales en términos de complejidad media, demostrando que cualquier otra solución se puede convertir en uno de estos árboles mediante modificaciones que conservan o disminuyen la complejidad media. El cálculo de la complejidad media correspondiente a estos árboles se inlcuye al final de este capítulo, pues en [15] se demuestra su optimalidad pero simplemente se señala el orden de complejidad que ofrecen en cuanto al coste medio de las operaciones.

Los árboles solución se pueden traducir a parejas de matrices dentro del *modelo algebraico* - a cada árbol le corresponde una pareja R, U de matrices - y al hacerlo se comprueba que el número de unos que suma la pareja de matrices que corresponde a uno de estos árboles solución es exactamente la cantidad que nosotros hemos obtenido trabajando directamente sobre matrices.

Por tanto, lo que presentamos en este capítulo es un método nuevo para establecer una cota inferior para el problema. En el camino obtenemos resultados relativos a matrices de ceros y unos que pueden ser de aplicación en otros problemas, y creemos que el método de trabajo con las matrices se podría utilizar para el estudio de las cotas inferiores en el problema más general que en esta tesis hemos denominado *problema de los rangos variables.*

La cota inferior para la complejidad media de las operaciones que obtenemos aplicando nuestro método de análisis es $\log_4 n + \frac{\log_4 n}{n} + \frac{1}{n}$ para un tamaño n del problema. Es también la complejidad media que ofrecen los árboles definidos en [15].

3.1 Introducción y Notación

A lo largo de este capítulo, R, U denotan matrices de ceros y unos de dimensión $n \times m$ y $m \times n$ respectivamente, $R = (r_{i,j})$, $U = (u_{i,j})$, con $R \times U = T$ siendo

$$T = (t_{i,j})_{i,j=1\ldots n}, \quad \text{con} \quad t_{i,j} = \begin{cases} 1 & i \geq j \\ 0 & i < j \end{cases}$$

Cuando sea necesario, especificaremos la dimensión de la matriz cuadrada T con un superíndice, esto es, T^n denota a la matriz T de dimensión $n \times n$.

Definimos

$$s_R = \sum_{i=1}^{n} \sum_{j=1}^{m} r_{i,j} \qquad s_U = \sum_{i=1}^{m} \sum_{j=1}^{n} u_{i,j}$$

Establecemos una cota inferior para la función

$$\phi(n,m) = min\{s_R + s_U\},$$

con R, U recorriendo el conjunto de matrices definido anteriormente.

Vamos a introducir notación y conceptos que nos serán necesarios a lo largo del capítulo.

Denotamos por $I^m_{i,j}$ a la matriz resultante de permutar las filas *i-ésima* y *j-ésima* de la matriz identidad de dimensión $m \times m$, a la que denotamos por I^m . Sabemos que para cualquier matriz A de dimensión $n \times m$, el efecto de multiplicar $A \times I^m_{i,j}$ es permutar las columnas *i-ésima* y *j-ésima* de A. De la misma forma, si A es de dimensión $m \times n$, entonces el efecto de multiplicar $I^m_{i,j} \times A$ es permutar las filas i, j de A. También sabemos que para cualquier i, j, m, $I^m_{i,j} \times I^m_{i,j} = I^m$.

Para cualquier matriz $X = (x_{i,j})_{i=1\ldots n,\ j=1\ldots m}$ denotamos,

$$\begin{aligned} X_{i,*} &= (x_{i,1}, x_{i,2}, \ldots, x_{i,m}) \\ X_{*,j} &= (x_{1,j}, x_{2,j}, \ldots, x_{n,j})^t \end{aligned}$$

Como resultado principal, en el Teorema 2 probamos que para cualquier pareja de matrices R, U,

$$\phi(n,m) \geq \mathrm{n}[\log_2 \mathrm{n}] + [\log_2 \mathrm{n}] + (\mathrm{m} - 2^{\mathrm{k}}) + 2 \qquad (2^k \leq n < 2^{k+1})$$

Usando nuestra notación, el problema de encontrar una cota inferior para la complejidad media de las operaciones *Update* y *Retrieve* consiste en establecer una cota inferior para la función $\phi(n,m)$ y dividirla entre $2n$, que es el número de operaciones distintas - n operaciones $Retrieve(i)$ y otras n operaciones $Update(j,\_)$, pues i, j toman valores entre 1 y n .

En la proposición 1 ofrecemos el cálculo de la complejidad media correspondiente a los árboles solución optimales definidos en [15].

Comenzamos con un lema algebraico que afirma que sólo tenemos que considerar el caso $n \leq m$, siendo n el tamaño del problema y m el número de variables utilizadas en la estructura de datos solución, que como sabemos coincide con el número de columnas de la matriz R y filas de la matriz U.

Lema 1

Si R, U tienen dimensión $n \times m$ and $m \times n$ respectivamente, entonces $n \leq m$.

Demostración

Sean r, u, t las aplicaciones lineales correspondientes a las matrices R, U, T respectivamente. Entonces,

$$R^n \xrightarrow{u} R^m \xrightarrow{r} R^n$$

De $T = R \circ U$, se deduce que $n = \dim(\mathrm{Im}(t)) = \dim(r(u(R^n))) \leq \dim(\mathrm{Im}(r)) \leq m$. Por tanto, necesariamente, $n \leq m$ ■

Observación 1 *Traducido al problema de las sumas parciales, el lema anterior afirma que dentro del modelo algebraico no existen soluciones que requieran menos de n variables, siendo n el tamaño del problema.*

3.2 Resultados Principales

Antes de establecer una cota inferior para la función ϕ, tenemos que probar algunos resultados.

La conclusión del Lema 2 y del Teorema 1 que se prueban a continuación es que, a efectos de establecer una cota inferior para el número de unos que hay en ambas matrices, podemos asumir sin pérdida de generalidad que las matrices R y U, en caso de que sean cuadradas, tienen todo unos en la diagonal principal y todo ceros por encima de dicha diagonal. En caso de que no sean cuadradas, tienen una forma que podemos denominar como *triangular escalonada*, similar a la de la pareja que ponemos como ejemplo en la Figura 1 a continuación:

$$\begin{pmatrix}
\mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & . & * & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & . & . & * & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & . & . & . & . & . & * & \mathbf{1} & 0 & 0 & 0 & 0 \\
* & . & . & . & . & . & . & * & \mathbf{1} & 0 & 0 & 0 \\
* & . & . & . & . & . & . & . & . & * & \mathbf{1} & 0 \\
* & . & . & . & . & . & . & . & . & . & * & \mathbf{1}
\end{pmatrix}
\qquad
\begin{pmatrix}
\mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\
. & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\
. & * & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\
. & . & * & \mathbf{1} & 0 & 0 & 0 & 0 \\
. & . & . & \mathbf{1} & 0 & 0 & 0 & 0 \\
. & . & . & \mathbf{1} & 0 & 0 & 0 & 0 \\
. & . & . & * & \mathbf{1} & 0 & 0 & 0 \\
. & . & . & . & * & \mathbf{1} & 0 & 0 \\
. & . & . & . & . & \mathbf{1} & 0 & 0 \\
. & . & . & . & . & * & \mathbf{1} & 0 \\
* & * & * & * & * & * & * & \mathbf{1}
\end{pmatrix}$$

Fig.1

El Lema 2 establece ciertas propiedades sobre grupos de columnas de R y las correspondientes filas de U. Partiendo de dichas propiedades, el Teorema 1 demuestra que mediante operaciones de intercambio de posición de columnas de R y las filas correspondientes de U - a través de multiplicaciones del tipo $R \times I_{i,j} \times I_{i,j} \times U$ que no afectan al número de unos - las matrices se pueden reordenar hasta adquirir la forma mencionada.

Lema 2 *Dadas* R, U, *de dimensiones respectivas* $n \times m$ *y* $m \times n$, *se tiene que para todo* $k \in \{1, \ldots, n\}$, *existe un conjunto no vacío de índices* $A_k = \{\sigma_k^1, \ldots, \sigma_k^{i_k}\} \subset \{1, \ldots, m\}$ *tal que*

(i) para todo $i \in A_k$, $u_{i,k} = 1$, *y para todo* $j > k$, $u_{i,j} = 0$

(ii) para todo $j \in A_k$,

a) $r_{i,j} = 0$ *si* $i < k$

b) existe $j_0 \in A_k$ *tal que* $r_{k,j_0} = 1$

c) $r_{k,j} = 0 \quad \forall j \in A_k, \quad j \neq j_0$

Demostración

Sin pérdida de generalidad, a efectos de establecer una cota inferior para el número de unos que hay en ambas matrices podemos asumir que toda columna de R tiene al menos un uno, pues en caso contrario, si tuviésemos $R_{*,j_0} = (0,\ldots,0)^t$, podríamos trabajar con nuevas matrices de dimensión $n \times (m-1)$, $(m-1) \times n$

$$R = (R_{*,1}, \ldots, R_{*,j_0-1}, R_{*,j_0+1}, \ldots, R_{*,m})$$

$$U = (U_{1,*}, \ldots, U_{j_0-1,*}, U_{j_0+1,*}, \ldots, U_{m,*})$$

De la misma forma, podemos asumir que todas las filas de U tienen al menos un uno.

También podemos asumir que en R no hay dos columnas iguales y en U no hay dos filas iguales, pues si tuviéramos $R_{*,j_0} = R_{*,j_1}$ $(j_0 < j_1)$, podríamos trabajar con

$$R = (R_{*,1}, \ldots, R_{*,j_0}, \ldots, R_{*,j_1-1}, R_{*,j_1+1}, \ldots, R_{*,m})$$

y

$$U = (U_{1,*}, \ldots, U_{j_0-1,*}, (U_{j_0,*} + U_{j_1,*}), U_{j_0+1,*}, \ldots U_{j_1-1,*}, U_{j_1+1,*}, \ldots, U_{m,*})$$

(obsérvese que si $R_{*,j_0} = R_{*,j_1}$, podemos concluir que para todo $j \in \{1,\ldots,n\}$ se verifica $u_{j_0,j} + u_{j_1,j} \leq 1$).

Y si lo que tuviéramos fuese $U_{j_0,*} = U_{j_1,*}$ $(j_0 < j_1)$, sería suficiente con considerar

$$R = (R_{*,1}, \ldots, R_{*,j_0-1}, (R_{*,j_0} + R_{*,j_1}), R_{*,j_0+1}, \ldots, R_{*,j_1-1}, R_{*,j_1+1}, \ldots, R_{*,m})$$

y

$$U = (U_{1,*}, \ldots, U_{j_0,*}, \ldots U_{j_1-1,*}, U_{j_1+1,*}, \ldots, U_{m,*})$$

Ahora procedemos con la demostración.

Sea $k \in \{1,\ldots,n\}$ y supongamos que no existe ningún índice que cumpla la condición *(i)*. Entonces, para todo $i \in \{1,\ldots,m\}$ tal que $u_{i,k} = 1$, existe $j \in \{k+1, k+2, \ldots, n\}$ con $u_{i,j} = 1$. Sea $i_0 \in \{1,\ldots,m\}$ tal que $u_{i_0,k} = 1$ (tal índice existe dado que U no puede tener una columna de ceros).

Sabemos que $R_{k,*} \times U_{*,k} = 1$, y por lo tanto existe z_0 tal que $r_{k,z_0} = 1 = u_{z_0,k}$. Pero estamos suponiendo que existe $j > k$ tal que $u_{z_0,j} = 1$, por lo que $R_{k,*} \times U_{*,j} = 1$, y hemos llegado a una contradicción. De esta forma, hemos probado por reducción al absurdo que existe un conjunto no vacío A_k que verifica la propiedad *(i)*.

Consideramos ahora dos situaciones distintas: si $A_k = \{j_0\}$, es decir el conjunto tiene un solo elemento, entonces $r_{i,j_0} = 0$ si $i < k$ y $r_{k,j_0} = 1$, y por tanto queda demostrado *(ii)*.

En otro caso, supongamos que existe otro índice j_1 tal que $j_0 \neq j_1$, con $j_0, j_1 \in A_k$; si tuviésemos que $r_{k,j_0} = 1 = r_{k,j_1}$, esto implicaría que $R_{k,*} \times U_{*,k} \geq 2$, pero esto es imposible, y dado que $r_{i,j_1} = 0$ si $i < k$ es trivialmente cierto, hemos probado *(ii)* ■

El siguiente resultado se sigue inmediatamente del Lema 2

Corolario 1

Dadas R, U *se tiene que para todo* $k \in \{1, \ldots, n\}$

$$0 < \sum_{i=1}^{m} u_{i,k} \leq m - k + 1$$

La siguiente figura puede darnos una idea intuitiva acerca del significado del Lema 2.

Sea un cierto $k \in \{0, \ldots, n\}$ y sean σ_k^l, σ_k^z, $\sigma_k^s \in A_k$,

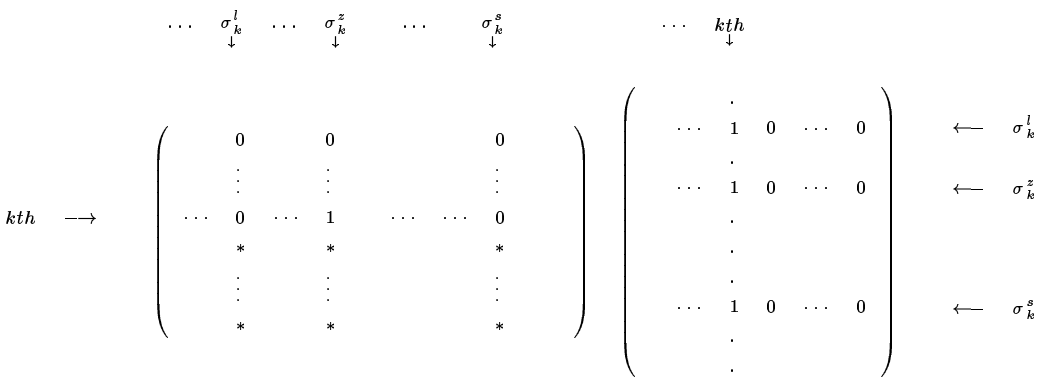


Fig.2

El siguiente teorema nos dice que podemos reordenar las columnas de R y las filas de U obteniendo parejas de matrices cuya forma es similar a la de la pareja mostrada en la Figura 1, que volvemos a mostrar a continuación:

$$\begin{pmatrix} \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & . & * & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & . & . & * & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & . & . & . & . & . & * & \mathbf{1} & 0 & 0 & 0 & 0 \\ * & . & . & . & . & . & . & * & \mathbf{1} & 0 & 0 & 0 \\ * & . & . & . & . & . & . & . & . & * & \mathbf{1} & 0 \\ * & . & . & . & . & . & . & . & . & . & * & \mathbf{1} \end{pmatrix} \qquad \begin{pmatrix} \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ . & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ . & * & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ . & . & * & \mathbf{1} & 0 & 0 & 0 & 0 \\ . & . & . & \mathbf{1} & 0 & 0 & 0 & 0 \\ . & . & . & \mathbf{1} & 0 & 0 & 0 & 0 \\ . & . & . & * & \mathbf{1} & 0 & 0 & 0 \\ . & . & . & . & * & \mathbf{1} & 0 & 0 \\ . & . & . & . & . & \mathbf{1} & 0 & 0 \\ . & . & . & . & . & * & \mathbf{1} & 0 \\ * & * & * & * & * & * & * & \mathbf{1} \end{pmatrix}$$

Fig.1

En la Figura 1, el cardinal de los distintos conjuntos A_k viene dado por $|A_1| = 1$, $|A_2| = 2$, $|A_3| = 1$, $|A_4| = 3$, $|A_5| = 1$, $|A_6| = 2$, $|A_7| = 1$ y $|A_8| = 1$. El cardinal de un cierto A_{i_0} determina la "*altura*"del salto entre la columna i_0 e $i_0 + 1$ de U y la "*anchura*"entre la fila i_0 e $i_0 + 1$ de R.

Teorema 1

Sin pérdida de generalidad, podemos asumir que las matrices R, U verifican las siguientes propiedades: existen $1 = j_1 < \ldots < j_n = m$, tales que para todo $k = 1 \ldots (n-1)$

- $u_{i,k} = 1$ *si* $j_k \leq i < j_{k+1}$, *y* $u_{i,k} = 0$ *si* $i < j_k$
- $u_{i,j} = 0$ *si* $j > k$
- $r_{k,j_k} = 1$ *y* $r_{k,j} = 0$ *si* $j > j_k$

Demostración

Para todo $k = 1 \ldots n$, sea σ_k^1 el único índice de A_k tal que $r_{k,\sigma_k^1} = 1$, y sea $i_k = |A_k|$.

Definimos

$$j_k = m - \sum_{s=k}^{n} i_s + 1 \quad \forall\ k = 1 \ldots n.$$

En la Figura 1, los valores correspondientes a estos índices serían : $j_1 = 1$, $j_2 = 2$, $j_3 = 4$, $j_4 = 5$, $j_5 = 8$, $j_6 = 9$, $j_7 = 11$ y $j_8 = 12$.

Para comprobar que $j_1 = 1$, basta con observar que $\sum_{s=1}^{n} i_s = m$, pues en otro caso, si tuviésemos $\sum_{s=1}^{n} i_s < m$, necesariamente tendría que existir $z_0 \in \{1 \ldots n\}$ tal que $R_{z_0,*} = (0 \ldots 0)^t$, y ya hemos asumido que esto no es posible.

Por otra parte, $j_n = m$ equivale a probar que $i_n = 1$; si fuese $i_n > 1$, implicaría que existen z_1, $z_2 \in A_n$, $z_1 \neq z_2$ tales que $u_{z_1,n} = 1 = u_{z_2,n}$, pero dado que $R \times U = T$, esto querría decir que o bien R_{*,z_1} es idénticamente cero o bien R_{*,z_2} es idénticamente cero, y partimos de que esto no es posible.

Sea $M_k = I_{\sigma_k^{i_k}, j_{k+1}-1} \times \ldots \times I_{\sigma_k^1, j_k}$ (obsérvese que eso implica que $M_k^{-1} = I_{\sigma_k^1, j_k} \times \ldots \times I_{\sigma_k^{i_k}, j_{k+1}-1}$).

Basta con darse cuenta de que si consideramos las matrices definidas como:

$$R^{'} = (\ldots((R \times M_n) \times M_{n-1}) \times M_1)$$

$$U^{'} = (M_1^{-1} \times (\ldots(M_{n-1}^{-1} \times (M_n^{-1} \times U))\ldots)$$

tendremos que $R^{'} \times U^{'} = T$, $m_{R'} = m_R$, $m_{U'} = m_U$, $s_{R'} = s_R$, $s_{U'} = s_U$.

Por otra parte, es evidente que las nuevas matrices $R^{'}$, $U^{'}$ verifican las condiciones requeridas ■

Los siguientes corolarios son trivialmente ciertos.

Corolario 2 *Si $m = n$, entonces podemos asumir que las matrices R, U son triangulares con ceros por encima de la diagonal principal.*

Corolario 3

$$\sum_{k=1}^{n}(r_{k,j_k} + u_{j_k,k}) + \sum_{k=1}^{n}\sum_{s=j_k+1}^{j_{k+1}-1} u_{s,k} = 2n + (m-n)$$

(el primer sumando cuenta los $2n$ unos que marcan el comienzo de cada uno de los "saltos" en las "diagonales-escalonadas" de las matrices R, U, y el segundo sumando cuenta el número total de unos que hay en los "escalones" de U, descontando los ya tenidos en cuenta en el primer sumando)

Observación 2 *Obsérvese que si* $m = n$, *simplemente hemos sumado las dos diagonales principales.*

Consideramos a partir de ahora que R, U son matrices de dimensiones respectivas $n \times m$, $m \times n$ siendo $m \geq n$ y $n = 2^k$ para un cierto número natural k. Una vez demostrado el teorema que establece la cota inferior para los valores n de la forma 2^k, extenderemos el resultado al caso general.

De momento pues, si tenemos $n = 2^k$, el Corolario 3 deberá decir

$$\sum_{k=1}^{n} \sum_{s=j_k+1}^{j_{k+1}-1} u_{s,k} + \sum_{k=1}^{n} (r_{k,j_k} + u_{j_k,k}) = 22^k + (m-n)$$

Los dos siguientes lemas nos ayudan en nuestro cálculo del número mínimo de unos que debe sumar una pareja R, U de matrices.

Comenzamos sumando las "subdiagonales" inmediatamente debajo de la diagonal principal en el Lema 3. Para ilustrar el significado de dicho Lema volvemos a la Figura 1. El Lema establece la cantidad de unos que tiene que haber en el conjunto de las posiciones marcadas en la figura con asterisco:

$$\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
. & * & * & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
. & . & . & * & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
. & . & . & . & * & * & * & 1 & 0 & 0 & 0 & 0 \\
. & . & . & . & . & . & . & * & 1 & 0 & 0 & 0 \\
. & . & . & . & . & . & . & . & * & * & 1 & 0 \\
. & . & . & . & . & . & . & . & . & . & * & 1
\end{pmatrix}
\qquad
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
* & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
. & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
. & * & 1 & 0 & 0 & 0 & 0 & 0 \\
. & . & * & 1 & 0 & 0 & 0 & 0 \\
. & . & . & 1 & 0 & 0 & 0 & 0 \\
. & . & . & 1 & 0 & 0 & 0 & 0 \\
. & . & . & * & 1 & 0 & 0 & 0 \\
. & . & . & . & * & 1 & 0 & 0 \\
. & . & . & . & . & 1 & 0 & 0 \\
. & . & . & . & . & * & 1 & 0 \\
 & . & . & . & . & . & * & 1
\end{pmatrix}$$

Fig.1

Lema 3 *Dadas* R, U *se tiene*

$$\sum_{k=1}^{2^k-1} \left[\sum_{j=j_k}^{j_{(k+1)}-1} r_{(k+1),j} + u_{j_{(k+1)},k} \right] = 2^k - 1 = 2^k - 2^1 + 1$$

Demostración

El resultado se sigue de que

$$R_{(k+1),*} \times U_{*,k} = 1 \qquad \forall k = 1, \ldots, n-1$$

y del hecho de que, por el Teorema 1, tenemos

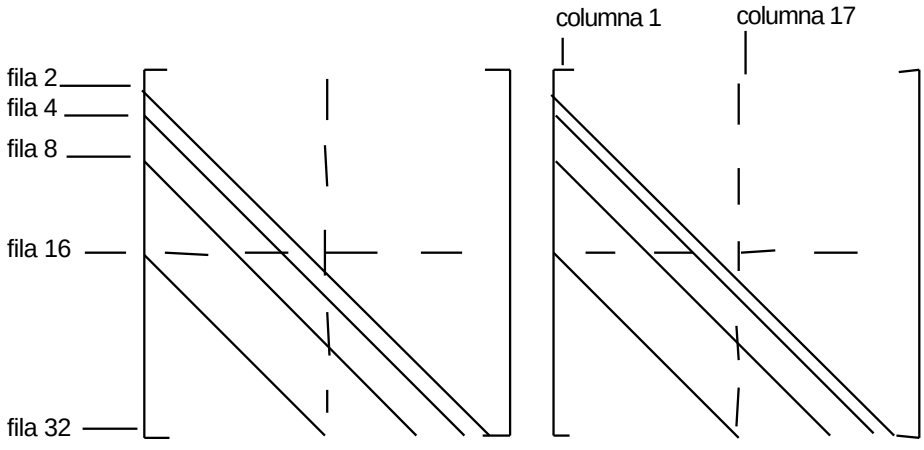
$$r_{(k+1),z} = \begin{cases} 1 & \text{si} \quad z = j_{(k+1)} \\ 0 & \text{si} \quad z > j_{(k+1)} \end{cases}$$

$$u_{z,k} = \begin{cases} 1 & \text{si} \ \ j_k \leq z < j_{(k+1)} \\ 0 & \text{si} \ \ z < j_k \end{cases}$$

para todo $k = 1 \ldots n-1$ ■

Previamente a la formulación y demostración del Lema 4, intentaremos dar una idea intuitiva del significado del mismo. Es más sencillo si imaginamos que las matrices R y U son cuadradas. En el Corolario 3 y el Lema 3 hemos contado respectivamente el número de unos presentes en las dos diagonales principales y en las dos subdiagonales inmediatamente por debajo. En el Lema 4 vamos a establecer una cota inferior para el número de unos considerando *bloques de subdiagonales* de distintos tamaños en ambas matrices.

En el ejemplo siguiente, se ve la división en bloques de subdiagonales consecutivas de distintos tamaños. La pareja de matrices tiene dimensión 32×32.

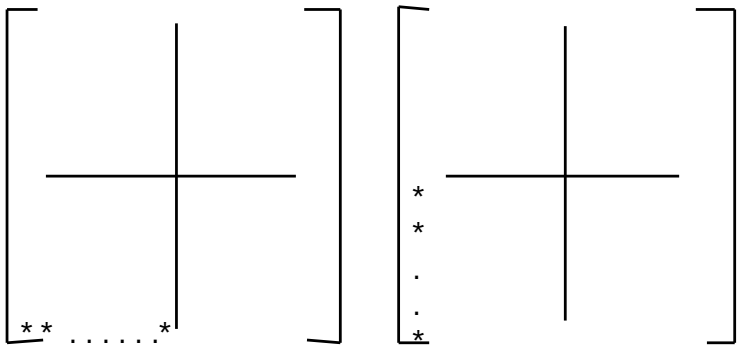


Nos centramos en el bloque de subdiagonales más ancho. Es fácil ver que dado que $R \times U = T$, o bien existe al menos un 1 en las posiciones 1 a 8 de la fila

16 de R - es decir, el trozo de fila incluído en el bloque de subdiagonales - o bien existirá obligatoriamente un 1 en la columna 1 de U en la zona correspondiente al bloque - es decir, filas 9 a 16. Emparejamos de manera consecutiva similar las siguientes filas del bloque en R con las columnas del bloque en U hasta llegar a la fila 32 y la columna 17 respectivamente, haciendo el mismo razonamiento. Por tanto, estos dos bloques de subdiagonales en R y U respectivamente, garantizan un mínimo de 17 unos. En el enunciado del Lema 4 a continuación, esta cantidad se obtendría para el valor del índice del sumatorio $i = 1$.

Se aplica la misma idea al resto de bloques hacia arriba, en este caso, los formados por 4 y 2 subdiagonales - se corresponden con los valores $i = 2 \ldots k-2$ en el enunciado del Lema 4 respectivamente. Siempre emparejamos una fila de R dentro del bloque con una columna de U, empezando por la primera fila de *anchura* máxima en el bloque de R y la columna 1 en el bloque de U.

En cuanto al valor del índice $i = 0$ en el enunciado del Lema 4, lo que hacemos es simplemente contar un 1 más, dado que si se observa la figura:



se verá fácilmente que o bien hay un 1 en la última fila de R, en la mitad marcada con asteriscos, o bien tendrá que haber al menos un 1 en la primera columna de U, en la mitad marcada con asteriscos - esto es así ya que $R \times U = T$.

El Lema 4 simplemente calcula la suma mínima de unos que garantizan todos los bloques. Tratamos con matrices que pueden no ser cuadradas, y eso hace que el cálculo resulte algo más complicado que en el ejemplo que acabamos de comentar.

Lema 4

Dadas R, U, *se verifica que para todo* $i = 0 \ldots k-2$

$$\sum_{z=0}^{2^k-2^{k-i}} \left(\sum_{s=j_{z+1}}^{j_{z+1+2^{k-(i+1)}}-1} r_{2^{k-i}+z,s} + \sum_{s=j_{z+1+2^{k-(i+1)}}}^{j_{z+2^{k-i}}} u_{s,z+1} \right) \geq 2^k - 2^{k-i} + 1$$

Demostración

Es suficiente con probar que para todo $z = 0 \ldots 2^k - 2^{k-i}$, se tiene:

$$\sum_{s=j_{z+1}}^{j_{z+1+2^{k-(i+1)}}-1} r_{2^{k-i}+z,s} + \sum_{s=j_{z+1+2^{k-(i+1)}}}^{j_{z+2^{k-i}}} u_{s,z+1} \geq 1 \qquad (1)$$

Supongamos que existe $t \in \{0 \ldots 2^k - 2^{k-i}\}$ tal que

$$\sum_{s=j_{t+1}}^{j_{t+1+2^{k-(i+1)}}-1} r_{2^{k-i}+t,s} = 0.$$

Entonces, por el Teorema 1 y el hecho de que $R_{2^{k-i}+t,*} \times U_{*,t+1} = 1$, se sigue que $\sum_{s=j_{z+1+2^{k-(i+1)}}}^{j_{(z+2^{k-i})}} u_{s,z+1} \geq 1$.

Aún tenemos que probar que cada posición de las matrices la estamos contando como mucho una vez en la expresión (1).

En primer lugar lo probamos para todas las filas de R : supongamos que existen v, $w \in \{0, \ldots, k-2\}$, $z_v \in \{0, \ldots, 2^k - 2^{k-v}\}$, $z_w \in \{0, \ldots, 2^k - 2^{k-w}\}$ tales que

$$2^{k-v} + z_v = 2^{k-w} + z_{w,} \quad v < w$$

Se deduce que

$$z_w - z_v = 2^{k-v} - 2^{k-w} \qquad (2)$$

Tenemos que probar que

$$j_{z_v+1+2^{k-(v+1)}} \leq j_{z_w+1}$$

pero dado que $x < y \Longrightarrow j_x < j_y$, es suficiente con probar

$$z_v + 1 + 2^{k-(v+1)} \leq z_w + 1$$

o equivalentemente

$$z_v + 2^{k-(v+1)} \leq z_w$$

Del hecho de que $v < w$, se sigue que

$$2^w \geq 2^{w-1} + 2^v$$

y por tanto

$$2^{k+w} - 2^{k+v} \geq 2^{k+w-1}$$

Dividiendo por 2^{v+w}, se obtiene

$$2^{k-v} - 2^{k-w} \geq 2^{k-(v+1)}$$

y aplicando (2), se deduce

$$z_v + 2^{k-(v+1)} \leq z_w$$

Ahora lo probamos para las columnas de U : sean $v, w \in \{0 \ldots k-2\}$, con $v < w$. Tenemos que probar que

$$j_{z+2^{k-w}} < j_{z+1+2^{k-(v+1)}},$$

pero, dado que j es estrictamente no decreciente, basta con probar

$$z + 2^{k-w} < z + 1 + 2^{k-(v+1)}$$

o lo que es lo mismo,

$$2^{k-w} - 2^{k-(v+1)} \leq 0$$

Multiplicando por 2^{w+v-k} tenemos

$$2^v - 2^{w-1} \leq 0$$

pero esto es trivialmente cierto dado que $v \leq w - 1$ ■

Ahora ya estamos en disposición de obtener la cota inferior que buscamos.

Teorema 2

Dados R, U, y un número natural k tal que $n = 2^k$,

$$\phi(n, m) \geq n \log_2 n + \log_2 n + (m - 2^k + 2)$$

Demostración

El teorema es consecuencia directa del Corolario 3, el Lema 3 y el Lema 4 ■

Obsérvese que si $2^k < n < 2^{k+1}$ para un cierto número natural k, la cota inferior que obtenemos es $n[\log_2 n] + [log_2 n] + (m - 2^k) + 2$, pues hay pocas cosas que cambiar en la demostración anterior. El Lema 4 debería decir

$$\sum_{z=0}^{n-2^{k-i}} \left(\sum_{s=j_{z+1}}^{j_{z+1+2^{k-(i+1)}}-1} r_{2^{k-i}+z,s} + \sum_{s=j_{z+1+2^{k-(i+1)}}}^{j_{z+2^{k-i}}} u_{s,z+1} \right) \geq n - 2^{k-i} + 1$$

y el Lema 3 quedaría como

$$\sum_{k=1}^{n-1} \left[\sum_{j=j_k}^{j_{k+1}-1} r_{k+1,j} + u_{j_{k+1},k} \right] = n - 1 = n - 2^1 + 1$$

Por otra parte, se observa que para todo $z = 1 \ldots (n - 2^k)$, $R_{2^k+z,*} \times U_{*,z+1} = 1$, por lo que $\sum_{z=1}^{n-2^k} [\sum_{s=1}^{j_{z+1+2^{k-1}}-1} (r_{2^k+z,s} + \sum_{l=j_{z+2^k-1}+1}^{j_{z+2^k}} u_{l,(z+1)})] \geq (n - 2^k)$ (obsérvese que $j_{z+1+2^{k-1}} - 1 \geq j_{z+2^{k-1}}$) ■

3.3 Cálculo de la Complejidad Media de los Árboles Optimales

Finalmente vamos a comprobar que la complejidad media que ofrecen los árboles definidos en [15] coincide con la cota inferior que hemos calculado. Como se mencionó en la introducción del capítulo, se sabe que dichos árboles son optimales en cuanto a la complejidad media de las operaciones, ya que se demuestra que cualquier solución al problema de tamaño n se puede modificar - con pasos que disminuyen o conservan la complejidad media - hasta hacerla coincidir con la solución descrita por el árbol solución para el problema de tamaño n.

Damos una breve descripción de los árboles definidos en [15] junto con los algoritmos que implementan las operaciones *Update* y *Retrieve* sobre dichos árboles. Por supuesto estas estructuras de datos solución tienen cabida en el

modelo algebraico definido por M.L.Fredman (ver [14] o el Capítulo 2 de esta tesis).

Para el problema de tamaño n el árbol correspondiente es un árbol binario completo con n nodos - completo salvo tal vez en el nivel más profundo, naturalmente.

Denotamos con $R_n^B,\ U_n^B$ a la pareja de matrices cuadradas de dimensión n inducidas por los árboles y algoritmos definidos en [15], y para todo número natural $n \geq 1$, llamamos $T(n)$ al número de unos que hay en $R_n^B,\ U_n^B$.

Para definir $R_n^B,\ U_n^B$, se debe considerar que los nodos del árbol están numerados de 1 a n en orden simétrico o *inorden* (es decir, de izquierda a derecha).

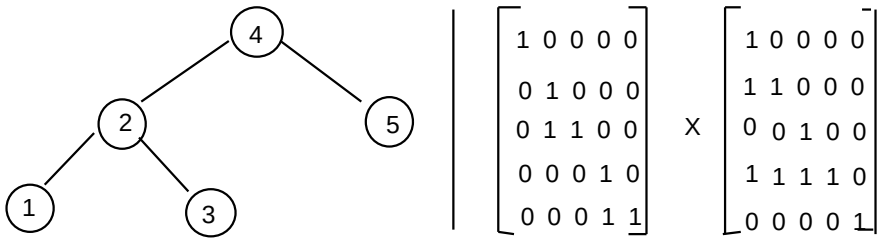
Sea z el *j-ésimo* nodo del árbol, y sean $z_0 = z, z_1, z_2 \ldots z_h, \quad h \geq 0$, los nodos a lo largo del camino que va desde z hasta la raíz de árbol, z_h.

Las correspondientes fila y columna *j-ésima* de $R_n^B,\ U_n^B$ respectivamente, se definen como:

$$r_{j,l}^B = 1 \longleftrightarrow [(l = j) \vee (\exists i = 1 \ldots h \ / \ (l = z_i \wedge \ z_{i-1} \ es\ el\ hijo\ derecho\ de\ z_i\ ))]$$

$$u_{l,j}^B = 1 \longleftrightarrow [(l = j) \vee (\exists i = 1 \ldots h \ / \ (l = z_i \wedge \ z_{i-1} \ es\ el\ hijo\ izquierdo\ de\ z_i\ ))]$$

Como ejemplo, mostramos el árbol asociado al problema de tamaño $n = 5$. En el interior de cada nodo aparece la numeración correspondiente al mismo en orden simétrico. Junto al árbol aparece la pareja de matrices $R,\ U$ correspondiente.



A continuación damos una definición recursiva de $R_n^B,\ U_n^B$ que nos ayudará

a calcular $T(n)$, y comprobaremos que para el conjunto de valores de la forma $n = 2^k$, el valor es $(\mathrm{n}\log_2 \mathrm{n} + \log_2 \mathrm{n} + 2)$, que es exactamente la cota inferior que hemos obtenido para $\phi(n, m)$ cuando $n = m$.

Nuestra definición recursiva es:

- para $n = 1$ la definición es trivial.
- para $n = 2$: $R_2^B = I^2$, $U_2^B = T^2$
- para $n > 2$: sea h el número correspondiente al nodo raíz. Entonces,

$$h = 2^{k-1} + (n - 2^k + 1) \qquad \forall n = 2^k \ldots ((3 \times 2^{k-1}) - 2)$$

$$h = 2^k \qquad \forall n = ((3 \times 2^{k-1}) - 1), \ldots (2^{k+1} - 1)$$

$$\mathrm{R}_n^B = \begin{bmatrix} \mathrm{R}_{h-1}^B & \begin{matrix}0\\0\\1\\1\end{matrix} & 0 \\ 0 & 0 & \mathrm{R}_{n-h}^B \end{bmatrix} \qquad \mathrm{U}_n^B = \begin{bmatrix} \mathrm{U}_{h-1}^B & \begin{matrix}0\\0\\1\\0\end{matrix} & 0 \\ 1 & 0 & \mathrm{U}_{n-h}^B \end{bmatrix}$$

Proposición 1

Para todo $n = 2^k$, $k = 0, 1 \ldots$

$$T(n) = n\log_2 n + log_2 n + 2$$

Demostración

En primer lugar, tenemos que probar el siguiente resultado:

$$T(n) = n[\log_2 n] + [\log_2 n] + (n - 2^k) + 2 + 2^k - 1 =$$

$$= (n + 1)[\log_2 n] + (n + 1) \qquad \forall\ n = 2^{k+1} - 1 \qquad (1)$$

Vamos con la prueba de (1).Utilizamos inducción.

Por definición de R_n^B, U_n^B cuando $n = 2^k - 1$, tenemos que

$$T(2^k - 1) = 2T(2^{k-1} - 1) + 2^k \qquad \forall k = 2, 3, \ldots$$

Para $n = 3$ el resultado es cierto, y asumimos que es cierto para $n \leq (2^k - 1)$.

Así pues,

$$T(n) = T(2^{k+1} - 1) = 2T(2^k - 1) + 2^{k+1} = 2[2^k(k-1) + 2^k] + 2^{k+1} =$$

$$= k2^{k+1} + 2^{k+1} = (n+1)[\log_2 n] + (n+1)$$

Ahora, procedemos a demostrar por inducción la proposición 1.

El resultado es cierto para $n = 2^1$, y lo asumimos cierto para $n = 2^2 \ldots 2^{k-1}$.

Por definición de $R^B_{2^k}$, $U^B_{2^k}$, se tiene que

$$T(2^k) = T(2^{k-1}) + T(2^{k-1} - 1) + (2^k + 1) \qquad (2)$$

Queremos probar

$$T(2^k) = 2^k k + k + 2 = (2^k + 1)k + 2$$

Por (2), y aplicando la hipótesis de inducción,

$$T(2^k) = (2^{k-1} + 1)(k-1) + 2 + T(2^{k-1} - 1) + (2^k + 1)$$

Y por (1),

$$T(2^k) = (2^{k-1} + 1)(k-1) + 2 + 2^{k-1}[\log_2(2^{k-1} - 1)] + 2^{k-1} + (2^k + 1) =$$

$$= k2^{k-1} + k + 1 + 2^{k-1}(k-2) + 2^k + 1 = k2^k + k + 2 =$$

$$= k(2^k + 1) + 2 \quad \blacksquare$$

Capítulo 4

Modelo Orbital para el Problema en el Caso Semigrupo

4.1 Un Nuevo Modelo de Cómputo para el Problema de las Sumas Parciales

Como ya hemos comentado en capítulos anteriores, el problema de las sumas parciales se ha estudiado enmarcándolo en distintos modelos. Entre ellos, y el que más ha servido de referencia para nuestro trabajo, el modelo algebraico de M.L.Fredman (ver [14] o el apartado 2.3 del Capítulo 2 de esta tesis). Recordamos que consiste en triplas $< R, U, Z >$ donde $Z = (z_1 \dots z_m)$ es un conjunto de variables, y R, U son matrices (de números enteros en el caso grupo o de ceros y unos en el caso semigrupo) cuyas filas y columnas sirven respectivamente para describir las operaciones de acceso y modificación. Las matrices deben verificar $R \times U = T$ donde, recordamos,

$$T = (t_{i,j})_{i,j=1\dots n}, \quad \text{siendo} \quad t_{i,j} = \begin{cases} 1 & i \geq j \\ 0 & i < j \end{cases}$$

En este capítulo trabajamos dentro del caso semigrupo. Definimos un mo-

delo nuevo, que utiliza dos vectores para describir las operaciones de acceso y modificación, en lugar de dos matrices. La estructura de datos solución en este caso es el propio vector que se desea mantener, es decir, los programas que implementan las operaciones operan sobre un vector de tamaño n.

Las ventajas de nuestro modelo son la simplicidad y el hecho de que se requiere menos espacio para definir las soluciones, en particular $\Theta(n)$ en lugar de $\Theta(n^2)$ bits (las matrices del modelo de Fredman tienen dimensión $n \times m$, $m \times n$ respectivamente, siendo m el número de variables que emplea la estructura de datos solución. En el Lema 1 del Capítulo 3 se demuestra que necesariamente $n \leq m$.

Además, podemos decir que nuestro modelo describe *soluciones efectivas* en el sentido de que es evidente una implementación inmediata de las soluciones - en el modelo matricial, los unos de una fila o columna determinan las variables a las que se deberá acceder, pero a la hora de escribir un programa que implemente la solución, ¡ habrá que pensar en alternativas que no obliguen a recorrer en cada caso una fila o columna buscando donde están los unos!

El modelo es más débil que el modelo matricial de M.L.Fredman, en el sentido de que incluye igualmente soluciones en las que una operación de acceso se efectúa sumando un subconjunto de las variables del vector, y una operación de modificación *Update* incrementando un subconjunto de dichas variables, pero no incluye *todas* esas posibles soluciones. Sin embargo, probaremos que están incluídas soluciones optimales en términos de la complejidad media de las operaciones. Nos referimos a soluciones optimales no sólo en cuanto al orden de complejidad, sino teniendo en cuenta los factores constantes: es decir, el número total de accesos a las variables del vector requerido por las $2n$ posibles operaciones, es estrictamente el mínimo requerido por cualquier solución del problema.

Nuestro modelo, al que llamaremos *modelo orbital* por razones que se explicarán a continuación, consiste en triplas $< A, B, V >$ donde A, B son vectores de tamaño n - tamaño del problema - que cumplen ciertas propiedades, junto con un vector V de tamaño n. Cada pareja de vectores representa una solución particular al problema de las sumas parciales, con su complejidad correspondiente.

Los algoritmos que implementan las operaciones de acceso y modificación se

definen a través de un conjunto de órbitas asociadas a dos aplicaciones lineales, de ahí el nombre de *modelo orbital*. Los valores que toman dichas aplicaciones lineales dependen de los valores almacenados en los dos vectores.

Es posible adaptar el modelo para describir soluciones al *problema de los rangos variables*, en el que la operación *Retrieve(i,j)* output $\sum_{k=i}^{j} v_k$, $1 \leq i \leq j \leq n$. Este tema se tratará en el apartado 4.2 al final de este capitulo.

4.1.1 Definición del Modelo Orbital para el Problema de las Sumas Parciales

Sea $< A^n, B^n, V >$ una tripla para el problema de las sumas parciales de tamaño n dentro del modelo orbital, con

$$A^n = (a_1 \ldots a_n), \;\; B^n = (b_1 \ldots b_n), \;\; V = (v_1 \ldots v_n)$$

Asumimos que A^n, B^n verifican las siguientes condiciones:

1. Propiedad estructural: A^n y B^n son vectores de dimensión n que almacenan números naturales, y tal que

$$1 \leq a_i \leq n - i + 1, \; 1 \leq b_i \leq i \quad \forall i = 1 \ldots n$$

2. Propiedad de órbita: dados $\mathcal{A}$, $\mathcal{B}$, α_k^n, β_k^n definidas como en la Definición 1,

   (a) para todo $j \leq i$, $\quad i, j \in \{1 \ldots n\}$, $\quad \#[\alpha_j^n \cap \beta_i^n] = 1$.

   (b) para todo $i < j$, $\quad i, j \in \{1 \ldots n\}$, $\quad \#[\alpha_j^n \cap \beta_i^n] = 0$.

Asumiendo estas propiedades damos las siguientes definiciones.

Definición 1

Dados $< A^n, \; B^n, \; V >$ *definimos las aplicaciones*

$$\mathcal{A} : \mathbf{N} \longrightarrow \mathbf{N} \qquad \mathcal{A}(k) = k + a_k$$

$$\mathcal{B} : \mathbf{N} \longrightarrow \mathbf{N} \qquad \mathcal{B}(k) = k - b_k$$

y las órbitas

$$\alpha_k^n = \{k, \mathcal{A}(k), \mathcal{A}^2(k), \ldots \mathcal{A}^l(k)\} \;\; con \;\; l = \max\{z / \mathcal{A}^z(k) \leq n\}$$

$$\beta_k^n = \{k, \mathcal{B}(k), \mathcal{B}^2(k), \ldots \mathcal{B}^l(k)\} \;\; con \;\; l = \max\{z / \mathcal{B}^z(k) \geq 1\}$$

(Para poder decir efectivamente que $\mathcal{A}$ y $\mathcal{B}$ son aplicaciones, asumiremos que $\mathcal{A}(k) = 0$ si $k > n$ y que $\mathcal{B}(k) = 0$ si $k < 1$).

Observación 3 *Obsérvese que como consecuencia de la propiedad estructural y las definiciones de $\mathcal{A}$, $\mathcal{B}$, α_k^n, β_k^n, podemos concluir que si*

$$\alpha_k^n = \{k, \mathcal{A}(k), \mathcal{A}^2(k), \ldots \mathcal{A}^{z_0}(k)\}, \qquad \beta_k^n = \{k, \mathcal{B}(k), \mathcal{B}^2(k), \ldots \mathcal{B}^{z_1}(k)\},$$

entonces

$$\mathcal{A}^{z_0+1}(k) = n+1, \;\; \mathcal{B}^{z_1+1}(k) = 0.$$

La Definición 2 a continuación, establece los algoritmos que implementan las operaciones update y retrieve .

Definición 2

Dados A^n, B^n, definimos los algoritmos que implementan las operaciones update y retrieve sobre el vector V como:

Update(j,x) *: para todo $z \in \alpha_j^n$, $v_z \leftarrow v_z + x$*

Retrieve(i) *: output $\sum_{z \in \beta_i^n} v_z$*

A continuación demostraremos que toda pareja de vectores que satisfaga las propiedades definidas anteriormente, representa una solución al *problema de las sumas parciales* sobre un vector V de longitud n, en la que las operaciones se implementan con los algoritmos que acabamos de definir.

Lema 5

Sean A^n, B^n vectores n-dimensionales que verifican la propiedad estructural y la propiedad de órbita. Afirmamos que dichos vectores representan una solución al problema de las sumas parciales sobre un vector V de longitud n, donde las operaciones de acceso y modificación se implementan a través de los algoritmos dados en la Definición 2.

Demostración

Nos inspiramos en el modelo de demostración utilizado en [14] para demostrar que una cierta implementación de las operaciones representa una solución correcta para el *problema de las sumas parciales.* La idea es que únicamente necesitamos considerar el efecto de efectuar consecutivamente las operaciones

$$\text{Retrieve(i), Update(j,x), Retrieve(i)}$$

para cualesquiera i, j, x.

Sean z_1, z_2 los output producidos como resultado de ejecutar las dos operaciones Retrieve(i), la primera y la segunda respectivamente.

A partir de la definición de las operaciones dada en la sección 4.1, es obvio que nuestros algoritmos representan una solución correcta si y sólo si

$$z_2 - z_1 = \begin{cases} x & j \leq i \\ 0 & i < j \end{cases}$$

pero esto es trivialmente cierto dado que los vectores A^n, B^n verifican la *propiedad de órbita* y a partir de la Definición 2 ■

Por último definimos la complejidad asociada a las operaciones de acceso y modificación dentro de nuestro modelo.

Definición 3

Dados dos vectores A^n, B^n, que representan una solución al problema de las sumas parciales dentro del modelo orbital, y dadas las correspondientes órbitas α_k^n, β_k^n con $k = 1 \ldots n$, definimos la complejidad de una operación $Update(j, x)$ como $\#\alpha_j^n$, y la complejidad de la operación $Retrieve(i)$ como $\#\beta_i^n$ (el símbolo $\#$ denota el cardinal de un conjunto).

4.1.2 Un Ejemplo Trivial

Como ejemplo, definimos una pareja de vectores que implementan la solución trivial que se deriva directamente a partir de la definición de las operaciones *Update* y *Retrieve* dada en el planteamiento general del problema de las sumas parciales (ver apartado 2.3.1 del Capítulo 2).

Definimos

$$\begin{aligned} A^n &= (a_i)_{i=1\ldots n} \text{ siendo } a_i = n - i + 1, \ \forall i \\ B^n &= (b_i)_{i=1\ldots n} \text{ siendo } b_i = 1 \ \forall i \end{aligned}$$

Las órbitas correspondientes serían

$$\begin{aligned} \alpha_k^n &= \{k\} \\ \beta_k^n &= \{k, k-1, \ldots 2, 1\} \ \forall k = 1 \ldots n \end{aligned}$$

De esta definición se deduce trivialmente que los vectores verifican la *propiedad de órbita.*

Esta solución conlleva una complejidad constante para cualquier operación *Update*, mientras que la complejidad de una operación *Retrieve(i)* depende linealmente del índice i.

4.1.3 El Modelo Incluye Soluciones Optimales en Complejidad Media

En este apartado construímos parejas de vectores de cualquier longitud que representan soluciones optimales al problema de las sumas parciales en el sentido de que la suma de los costes de las $2n$ operaciones distintas es mínima.

En [15] se definen árboles binarios que constituyen soluciones optimales al problema de las sumas párciales en cuanto a la complejidad media de las operaciones. Hemos descrito brevemente estos árboles en el apartado 3.3 del Capítulo 3 de esta tesis.

Basaremos la construcción de nuestros vectores solución en acomodar dentro de nuestro modelo dichos árboles binarios.

En el siguiente teorema definimos los vectores y probamos su optimalidad

Teorema 3

El modelo orbital incluye soluciones optimales para el problema de las sumas parciales en cuanto a la complejidad media de las operaciones.

Demostración

Comenzamos definiendo los vectores A^n, B^n que representan tal tipo de solución.

Damos una definición recursiva:

- $n = 1 : A^1 = (1), \quad B^1 = (1)$
- $n = 2 : A^2 = (1,1), \quad B^2 = (1,2)$
- $n > 2$: sea $z = [\frac{n+1}{2}]$. Entonces,

$$A_k^n = \begin{cases} A_k^{z-1} & \text{si } 1 \leq k < z \\ n - z + 1 & \text{si } k = z \\ A_{k-z}^{n-z} & \text{si } z < k \leq n \end{cases}$$

$$B_k^n = \begin{cases} B_k^{z-1} & \text{si } 1 \leq k < z \\ z & \text{si } k = z \\ B_{k-z}^{n-z} & \text{si } z < k \leq n \end{cases}$$

Lo primero que debemos hacer es comprobar que dichos vectores están dentro del *modelo orbital*, es decir, debemos probar que satisfacen la *propiedad de órbita* pues la *propiedad estructural* la satisfacen trivialmente.

Por la definición de las órbitas dada en 1 y por la *propiedad estructural*, podemos afirmar

1. $\alpha_z^n = \{z\}, \ \beta_z^n = \{z\}$
2. $\alpha_j^n = \alpha_{j-z}^{n-z} \ \forall j > z$
3. $\beta_i^n = \beta_i^{z-1} \ \forall i < z$
4. $\alpha_j^n = \alpha_j^{z-1} \cup \{z\} \ \forall j < z$
5. $\beta_i^n = \beta_{i-z}^{n-z} \cup \{z\} \ \forall i > z$

Una vez establecidos estos puntos, verificamos que para cualesquiera α_j^n, β_i^n se verifica la *propiedad de orbita*:

- $i, j \leq z$: aplicando inducción y el punto 4
- $i, j > z$: por inducción y por 5

- $(i = z \wedge j \neq z)$: a partir de 1, 2 y 4
- $(i \neq z \wedge j = z)$: se deduce de 1, 3 y 5
- $(i > z \wedge j < z)$: es trivialmente cierto que $(\alpha_j^n \cap \beta_i^n) = \{z\}$
- $(i < z \wedge j > z)$: la intersección es vacía trivialmente

A continuación verificamos que la solución que representan dichos vectores es optimal en términos de complejidad media.

Dados A^n, B^n, llamamos

$$\phi(n) = \sum_{k=1}^{n} (|\alpha_k^n| + |\beta_k^n|).$$

Esto es, $\phi(n)$ denotal el tiempo total de ejecución correspondiente a las $2n$ operaciones. La complejidad media viene dada, por tanto, por la expresión $\phi(n)/2n$.

A partir de la definición recursiva de A^n, B^n tenemos que

$$\phi(n) = n + 1 + \phi(n - z) + \phi(z - 1) \text{ siendo } z = \left[\frac{n+1}{2}\right]$$

.

Como casos base de la inducción sabemos que $\phi(1) = 2$, $\phi(2) = 5$.

Dependiendo de que la longitud n de los vectores sea par o impar, distinguimos dos casos

- n par: $z = \frac{n}{2} \Rightarrow z - 1 = \frac{n-2}{2}$, $n - z = \frac{n}{2}$
- n impar: $z = \frac{n+1}{2} \Rightarrow z - 1 = \frac{n-1}{2}$, $n - z = \frac{n-1}{2}$

En primer lugar calculamos la diferencia entre los valores de la función ϕ para dos valores consecutivos de los argumentos, el primero impar y el segundo impar

$$\phi(2k+1) - \phi(2k) = 2k+2+2\phi(k) - (2k+1+\phi(k)+\phi(k-1)) = 1+(\phi(k)-\phi(k-1))$$

y lo mismo pero ahora el primer argumento es un par y el segundo su impar inmediatamente anterior

$$\phi(2k+2) - \phi(2k+1) = 2k+3+(\phi(k+1)+\phi(k)) - (2k+2+2\phi(k)) = 1+(\phi(k+1)-\phi(k))$$

Ya podemos generalizar para un n cualquiera,

$$(\phi(n+1) - \phi(n)) = 1 + (\phi(z) - \phi(z-1))$$

Nos ayudamos de una función auxiliar h que definimos como

$$h(n) = [n/2]\,.$$

Tenemos que

$$\begin{aligned}\phi(n+1) - \phi(n) &= 1 + [\phi(h(n+1)) - \phi(h(n+1)-1)] = 1 + [1 + \{\phi(h^2(n+1)) - \phi(h^2(n+1)-1)\}] \\ &= \ldots = 1 + 1 + \ldots + 1(\phi(2) - \phi(1)) = k + \phi(2) - \phi(1) = k + 3\end{aligned}$$

para un cierto k que procedemos a calcular.

Estamos buscando el valor k tal que $h^k(n+1) = 2$.

Tenemos

$$h^k(n+1) = \left[\frac{h^{k-1}(n+1)}{2}\right] = 2$$

y eso significa

$$\begin{aligned}4 \leq h^{k-1}(n+1) < 6 \quad &\Rightarrow \quad 2^2 \leq \left[\frac{h^{k-2}(n+1)}{2}\right] < 3\ .\ 2 \ \Rightarrow\ 2^3 \leq h^{k-2}(n+1) < 3\ .\ 2^2 \\ &\Rightarrow \quad \ldots \ \Rightarrow\ 2^{k+1} \leq n+1 < 3\ .\ 2^k \\ &\Rightarrow \quad k+1 \leq \log_2(n+1) < k + \log_2 3\end{aligned}$$

y de aquí concluímos que $k = [\log_2(n+1) - 1]$.

Así que de momento,

$$\phi(n+1) - \phi(n) = k + 3, \text{ con } k = [\log_2(n+1) - 1]$$

Por tanto,

$$\begin{aligned}\phi(n+1) &= k + 3 + \phi(n) = [\log_2(n+1) - 1] + 3 + \phi(n) \\ &= (3+3) + \phi(n-1) + ([\log_2(n+1) - 1] + [\log_2(n) - 1]) \\ &= \phi(1) + 3n + \sum_{j=2}^{n+1} [\log_2 j - 1]\end{aligned}$$

Calculamos en primer lugar

$$\begin{aligned}\sum_{j=2}^{n+1}[\log_2 j-1] &= \sum_{j=4}^{n+1}[\log_2 j-1] = 4+\sum_{j=8}^{n+1}[\log_2 j-1] = 4\,.\,1+8\,.\,2+\sum_{j=16}^{n+1}[\log_2 j-1]=\ldots \\ &= 2^2\,.\,1+2^3\,.\,2+\ldots+2^{[\log_2(n+1)]-1}([\log_2(n+1)]-2)+ \\ &\quad +(n+2-2^{[\log_2(n+1)]})([\log_2(n+1)]-1)\end{aligned}$$

Probaremos después el siguiente resultado

$$2^2\,.\,1+2^3\,.\,2+\ldots+2^{[\log_2(n+1)]-1}([\log_2(n+1)]-2) = ([\log_2(n+1)]-3)2^{[\log_2(n+1)]}+4 \qquad (1)$$

Asumiendo dicho resultado, tenemos que

$$\begin{aligned}\phi(n+1) &= \phi(1)+3n+([\log_2(n+1)]-3)2^{[\log_2(n+1)]}+4+n[\log_2(n+1)]-n+ \\ &\quad +2[\log_2(n+1)]-2-[\log_2(n+1)]2^{[\log_2(n+1)]}+2^{[\log_2(n+1)]} \\ &= 4+(n+2)[\log_2(n+1)]+2n-2(2^{[\log_2(n+1)]})\end{aligned}$$

Esto nos da un tiempo total de ejecución para las $2n$ operaciones de

$$\phi(n) = 2+2n+(n+1)[\log_2(n)]-2(2^{[\log_2(n)]})$$

La complejidad media viene dada por $\phi(n)/2n$, y eso significa que tomando k tal que $2^k \leq n < 2^{k+1}$ tenemos

$$\phi(n) = 2+2n+(n+1)k-2(2^k) \Rightarrow \frac{\phi(n)}{2n} = \frac{1}{n}+1+\frac{k}{2}+\frac{k}{2n}-\frac{2^k}{n}, \quad k=[\log_2 n]$$

y con esto la demostración está terminada, pues esta es exactamente la cota inferior optimal para la complejidad media de las operaciones que conocemos por dos caminos distintos :

- como consecuencia de la probada optimalidad en el caso medio de los árboles binarios definidos en [15]. Su complejidad asociada exacta, coincidente con la que acabamos de obtener, aparece calculada en el apartado 3.3 del Capítulo 3 de esta tesis.

- a través de la demostración directa que se presenta como resultado en el Capítulo 3 de esta tesis (véase el Teorema 2)

Nos quedaba probar la igualdad referida con (1) mas arriba.

Si llamamos $r = [\log_2(n+1)] - 2$, tenemos

$$2^2 \,.\, 1+2^3 \,.\, 2+\ldots+2^{[\log_2(n+1)]-1}([\log_2(n+1)]-2) = \sum_{l=1}^{r} l \,.\, 2^{l+1} = (2^{r+2}-2^2)+\sum_{l=2}^{r}(l-1) \,.\, 2^{l+1}$$

Así pues,

$$\begin{aligned}
\sum_{l=1}^{r} l \,.\, 2^{l+1} &= (2^{r+2}-2^2)+\sum_{l=2}^{r}(l-1) \,.\, 2^{l+1} = (2^{r+2}-2^2)+(2^{r+2}-2^3)+\sum_{l=3}^{r}(l-2) \,.\, 2^{l+1} = \ldots \\
&= (2^{r+2}-2^2)+(2^{r+2}-2^3)+(2^{r+2}-2^4)+\ldots+(2^{r+2}-2^r)+2^{r+1} \\
&= (r-1)2^{r+2}-(2^{r+1}-2^2)+2^{r+1} \\
&= (r-1)2^{r+2}+2^2 = ([\log_2(n+1)]-3) \,.\, 2^{[\log_2(n+1)]}+4
\end{aligned}$$

y hemos completado la demostración ■

4.2 Adaptación del Modelo Orbital al Problema de los Rangos Variables

A continuación adaptamos el *modelo orbital* al problema de los rangos variables en el caso semigrupo.

Recordamos la formulación del *problema de los rangos variables:* se desea mantener un vector V de longitud n que almacena valores de un semigrupo conmutativo arbitrario, sobre el que se realizan operaciones de acceso y modificación - *Retrieve* y *Update* respectivamente - definidas por

(a) *Update(i, x):* $V[i] \leftarrow v_i + x$, $\quad 1 \leq i \leq n,\ x \in G$

(b) *Retrieve(i, j):* output $\sum_{k=i}^{j} v_k \quad 1 \leq i \leq j \leq n$

Utilizaremos dos vectores para describir las operaciones que se efectuarán sobre el vector V. La longitud de ambos vectores es $\frac{n(n+1)}{2}$. Las operaciones se ejecutarán sobre un vector $\overline{V}$ de longitud $\frac{n(n+1)}{2}$.

De nuevo los algoritmos que implementan las operaciones de acceso y modificación se describen a través de un conjunto de órbitas asociadas a dos aplicaciones lineales.

Para que los vectores representen una solución al problema de los rangos variables, deben cumplir ciertas propiedades.

Cada pareja de vectores representa una solución particular del problema, con su correspondiente complejidad.

Este modelo incluye una clase de soluciones que reutiliza los vectores solución del *modelo orbital para el problema de las sumas parciales* que hemos definido en los apartados anteriores. La idea se basa en el hecho de que el conjunto de las $\frac{n(n+1)}{2}$ posibles operaciones $Retrieve(i,j)$ se puede dividir en n conjuntos $R_1 \ldots R_n$, tales que las operaciones de cada conjunto R_z son $\{Retrieve(z,s)/z \leq s \leq n\}$.

Probaremos que el modelo incluye parejas de vectores solución cuya complejidad media asociada es $\Theta(\log n)$.

Como referencia, la complejidad media asociada a la solución trivial del problema dada por los programas

(a) *Update(i, x):* $V[i] \leftarrow v_i + x$

(b) *Retrieve(i, j):* output $\sum_{k=i}^{j} v_k$

es $\Theta(n)$, pues la suma de las complejidades de las $n + \frac{n(n+1)}{2}$ operaciones posibles es

$$\begin{aligned} n + \sum_{i=1}^{n} \sum_{j=1}^{i} j &= n + \sum_{i=1}^{n} \frac{i(i+1)}{2} \\ &= n + \frac{1}{2}\frac{n(n+1)}{2} + \frac{1}{2}\sum_{i=1}^{n} i^2 \\ &\in \Theta(n^3) \end{aligned}$$

A continuación definimos formalmente el modelo.

4.2.1 Definicion del Modelo Orbital para el Problema de los Rangos Variables

Consideramos el problema de los rangos variables de tamaño n. Sean

$$U^M = (u_1 \ldots u_M), \quad W^M = (w_1 \ldots w_M) \quad \text{con} \quad M = \frac{n(n+1)}{2},$$

los vectores que definen los algoritmos que implementan dichas operaciones sobre la estructura de datos solución, el vector $\overline{V}$ de longitud $\frac{n(n+1)}{2}$.

Asumimos que U^M, W^M verifican la siguiente *propiedad estructural*: sea $z_i = \sum_{j=0}^{i-1}(n-j)$, $i = 0 \ldots n$; los vectores U^M, W^M se definen de forma que para todo $i = 0 \ldots n-1$, se tiene que

$$< U^M_{z_i+1 \ldots z_{i+1}}, W^M_{z_i+1 \ldots z_{i+1}} >$$

representa una pareja de vectores solución para el *problema de las sumas parciales* de tamaño $(n-i)$, dentro del *modelo orbital* definido para dicho problema.

Asumiendo que los vectores cumplen dicha *propiedad estructural*, se tienen las siguientes definiciones.

Definición 4

Dados $\overline{V}$, U^M, W^M *y dados* $z_0 \ldots z_n$ *con* $z_i = \sum_{j=0}^{i-1}(n-j)$, *definimos las aplicaciones*

$$\mathcal{A} : \mathbf{N} \longrightarrow \mathbf{N} \qquad \mathcal{A}(k) = k + u_k$$

$$\mathcal{B} : \mathbf{N} \longrightarrow \mathbf{N} \qquad \mathcal{B}(k) = k - w_k$$

y las órbitas

$\alpha^M_k = \{k, \mathcal{A}(k), \mathcal{A}^2(k), \ldots \mathcal{A}^l(k)\}$ *siendo* $l = \max\{q / \mathcal{A}^q(k) \leq z_{i+1}\}$, $k = z_i{+}1 \ldots z_{i+1}$

$\beta^M_k = \{k, \mathcal{B}(k), \mathcal{B}^2(k), \ldots \mathcal{B}^l(k)\}$ *siendo* $l = \max\{q / \mathcal{B}^q(k) \geq z_i{+}1\}$, $k = z_i{+}1 \ldots z_{i+1}$

La siguiente definición presenta los programas que implementan las operaciones de acceso y modificación.

Definición 5

Dados U^M, W^M, *definimos los algoritmos que implementan las operaciones Update y Retrieve sobre* $\overline{V} = (\overline{v_1}, \overline{v_2} \ldots)$ *como:*

Update(j,x) *: para todo* $l \in \overline{\alpha^M_j}$, $\overline{v_l} \leftarrow \overline{v_l} + x$, *siendo*

$$\overline{\alpha^M_j} = \bigcup_{s=0}^{j-1} \alpha^M((j-s) + z_s)$$

Retrieve(i,j) : *output* $\sum_{l \in \beta_s^M} \overline{v_l}$, *con* $s = z_{i-1} + (j - i + 1)$

A continuación demostraremos que toda pareja de vectores que satisfaga la *propiedad estructural* definida anteriormente, representa una solución al *problema de los rangos variables* de tamaño n que utiliza como estructura de datos solución $\frac{n(n+1)}{2}$ variables organizadas en un vector $\overline{V}$, y en la que las operaciones se implementan con los algoritmos que acabamos de definir.

Lema 6

Sean U^M, W^M *vectores* M*-dimensionales,* $M = \frac{n(n+1)}{2}$ *, que verifican lo establecido en la propiedad estructural enunciada al comienzo de la sección 4.2.1 . Afirmamos que dichos vectores representan una solución al problema de los rangos variables de tamaño* n*. Las operaciones de acceso y modificación se implementan a través de los algoritmos dados en la Definición 5, y operan sobre* $\frac{n(n+1)}{2}$ *variables organizadas en un vector* $\overline{V}$*.*

Demostración

De nuevo, nuestra demostración sigue el modelo de demostración utilizado en [14]. En este caso, necesitamos considerar el efecto de efectuar consecutivamente las operaciones

Retrieve(i,j), Update(r,x), Retrieve(i,j)

para cualesquiera $i, j, r \in \{1 \ldots n\}$, x elemento del semigrupo conmutativo cuyos valores se almacenan en V.

Sean q_1, q_2 los output producidos como resultado de ejecutar las dos operaciones Retrieve(i,j), la primera y la segunda respectivamente.

A partir de la definición de las operaciones dada en el planteamiento general del problema (ver comienzo del apartado 4.2), es obvio que nuestros algoritmos representan una solución correcta si y sólo si

$$q_2 - q_1 = \begin{cases} x & i \leq r \leq j \\ 0 & \text{e.o.c.} \end{cases}$$

Para la demostración, tendremos presentes las definiciones de órbitas y programas dadas respectivamente en las Definiciones 4 y 5.

Tenemos que probar que:

$$\#[\overline{\alpha_r^M} \cap \beta_k^M] = \begin{cases} 1 & i \leq r \leq j \\ 0 & \text{e.o.c.} \end{cases}$$

siendo $k = z_{i-1} + (j - i + 1)$.

Analizamos las órbitas correspondientes a las dos operaciones:

- *Retrieve*(i, j): por la definición del programa que implementa la operación, sabemos que la órbita afectada es β_k^M, con $k = z_{i-1} + (j - i + 1)$. Dado que, por definición de los valores z_i, tenemos $z_i - z_{i-1} = (n - i + 1)$, podemos afirmar que $k \in z_{i-1} + 1 \ldots z_i$ (los argumentos i, j, z de las operaciones *Retrieve*(i, j), *Update*(z), son valores entre 1 y n).

  La órbita β_k^M, está formada por un conjunto de valores, el mayor de los cuales es el propio k, y el menor de los cuales es necesariamente mayor o igual que $z_{i-1} + 1$.

- *Update*(r):

  1. $(r < i)$: la órbita asociada a la operación, $\overline{\alpha_r^M}$, es una unión de órbitas, de las cuales la que incluye los valores mayores es $\alpha^M(1 + z_{r-1})$ (obsérvese que $z_i > i \; \forall i = 1 \ldots n$). Pero el mayor valor contenido en esa órbita es menor o igual que z_r y por tanto estrictamente menor que $z_{i-1} + 1$. Concluímos que la intersección de las órbitas correspondientes a *Retrieve*(i, j) y *Update*(r, x) es vacía, lo que nos da el resultado buscado.

  2. $(r > j)$: tenemos que analizar los valores comprendidos en la órbita $\alpha^M((r - (i - 1)) + z_{i-1})$. El menor valor contenido en dicha órbita es precisamente $(r - (i - 1)) + z_{i-1}$, pero dicho valor es estrictamente mayor que $(j - (i - 1)) + z_{i-1}$. De nuevo concluímos que la intersección de las órbitas correspondientes a las operaciones *Retrieve*(i, j) y *Update*(r, x) es vacía y por tanto el resultado a probar es cierto.

  3. $(i \leq r \leq j)$: observamos que $z_{i-1} + 1 \leq ((j - i + 1) + z_{i-1})$, $((r - i + 1) + z_{i-1}) \leq z_i$. El resultado es trivialmente cierto debido a que $< U_{z_{i-1}+1 \ldots z_i}, W_{z_{i-1}+1 \ldots z_i} >$ son vectores solución para el *problema*

de la sumas parciales dentro del *modelo orbital* definido para dicho problema, y por tanto cumplen la *propiedad de órbita*, lo que nos garantiza que el cardinal de la intersección de nuestras orbitas es 1 ■

Por último definimos la complejidad asociada a las operaciones de acceso y modificación dentro de nuestro modelo.

Definición 6

Dados dos vectores U^M, W^M, *que representan una solución al problema de los rangos variables dentro del modelo orbital definido para tal problema, y dadas las correspondientes órbitas* $\overline{\alpha_k^M}$, β_k^M, *definimos la complejidad de una operación* $Update(j,x)$ *como* $\#\overline{\alpha_j^M}$, *y la complejidad de la operación* $Retrieve(i,j)$ *como* $\#\beta_k^M$, *siendo*

$$k = (j - i + 1) + z_{i-1}, \qquad z_i = \sum_{j=0}^{i-1}(n - j), \ \ \forall i = 0 \ldots n$$

(el símbolo # denota el cardinal de un conjunto).

4.2.2 El Modelo Incluye Soluciones Logarítmicas en Complejidad Media

Probaremos a continuación que el modelo incluye soluciones cuya complejidad media asociada es $\Theta(\log n)$.

Teorema 4

El modelo orbital para el problema de los rangos variables incluye soluciones cuya complejidad media es $\Theta(\log n)$.

Demostración

Para construir tales soluciones, utilizaremos los vectores A^n, B^n definidos en 3, que representan soluciones optimales (complejidad media) para el *problema de las sumas parciales* de tamaño n. α_k^n, β_k^n (superíndice en letra minúscula) denotan las órbitas correspondientes a las operaciones.Recordamos que la suma

total de las complejidades asociadas a las $2n$ operaciones distintas, era de $2 + 2n + (n+1)k - 2 \cdot 2^k$ para $2^k \leq n < 2^{k+1}$.

Definimos los vectores solución U^M, W^M para el *problema de los rangos variables* como:

$$< U^M_{z_i+1 \ldots z_{i+1}}, W^M_{z_i+1 \ldots z_{i+1}} > \; = \; < A^{n-i}, B^{n-i} > \qquad i = 0 \ldots n-1$$

Esto, en primer lugar, garantiza que U^M, W^M se encuentran dentro del *modelo orbital* para el problema de los rangos variables

Llamamos $z_i = \sum_{j=0}^{i-1}(n-j) \quad i = 0 \ldots n$.

Estudiamos la complejidad asociada a las operaciones *Retrieve* y *Update*.

- *Retrieve*(i,j): por la definición de los algoritmos que implementan las operaciones - Definición 5 - sabemos que la órbita asociada a la operación es β^M_k, siendo $k = (j-i) + 1 + z_{i-1}$.

  Por definición de órbitas y por la forma en que hemos definido U^M, W^M tenemos que $\quad \#\beta^M_k = \#\beta^{n-(i-1)}_{k-z_{(i-1)}} \qquad k = z_{i-1}+1 \ldots z_i$

  Por tanto,

  $$\sum_{k=1}^{\frac{n(n+1)}{2}} \#\beta^M_k \; = \; \sum_{i=1}^{n} \sum_{k=z_{i-1}+1}^{z_i} \#\beta^{n-(i-1)}_{k-z_{(i-1)}} \; = \; \sum_{i=1}^{n} \sum_{k=1}^{n-(i-1)} \#\beta^{n-(i-1)}_k$$

- *Update*(r,x): por la definición de los algoritmos que implementan las operaciones - Definición 5 - sabemos que la órbita asociada a la operación es

  $$\overline{\alpha^M_r} = \bigcup_{i=0}^{r-1} \alpha^M_{((r-i)+z_i)}$$

  Por definición de órbitas y de los vectores U^M, W^M, tenemos que

  $$\# \bigcup_{r=1}^{n} \alpha^M_{((r-i)+z_i)} \; = \; \# \bigcup_{r=1}^{n-i} \alpha^{n-i}_r \qquad i = 0 \ldots (r-1)$$

En conclusión, si llamamos

$$\phi(n) \; = \; \sum_{k=1}^{n} (|\alpha^n_k| + |\beta^n_k|)$$

$$\psi(n) \; = \; \sum_{k=1}^{n} |\overline{(\alpha^M_k}| + \sum_{k=1}^{\frac{n(n+1)}{2}} |\beta^M_k| \qquad \text{siendo} \quad M = \frac{n(n+1)}{2},$$

tenemos que

$$\psi(n) = \sum_{i=1}^{n} \phi(n)$$

En el apartado 5.1.1 del Capítulo 5 de esta tesis se demuestra que $\sum_{i=1}^{n} \phi(n) \in \Theta(n^2 \log_n)$, por lo que no incluímos aquí el cálculo.

Por tanto, dado que la complejidad media asociada a las operaciones definidas por los vectores U^M, W^M viene dada por $\frac{\psi(n)}{n+\frac{n(n+1)}{2}}$, deducimos que dicha complejidad media es $\Theta(\log_n)$ ■

Capítulo 5

Modelos de Cómputo y Soluciones para el Problema de los Rangos Variables

El planteamiento general del problema de los rangos variables y el marco formal para el estudio del problema establecido en [19] - al que haremos referencia en este capítulo - se puede consultar en el apartado 2.3.2 del Capítulo 2.

En este capítulo

- Definimos un modelo matricial equivalente al descrito en [19], en el cual enmarcar el estudio del *problema de los rangos variables.*
- Reutilizamos el conocimiento de la cota inferior optimal para la complejidad media de las operaciones en el *problema de las sumas parciales* con el fin de obtener una cota inferior $\Omega(\log n)$ para la suma de la complejidad media de las operaciones *Retrieve* y la complejidad media de las operaciones *Update* (considerando la complejidad media de ambas operaciones por separado) en el problema de los rangos variables de tamaño n. Este resultado ya es conocido (ver *Teorema 1* de [19]) si nos referimos al orden de complejidad, pero con nuestro método obtenemos mejores constantes en la expresión de la función - nuestra cota inferior es *mayor.*

- Definimos soluciones dentro de nuestro modelo matricial, que ofrecen una complejidad media de orden constante *para el conjunto de todas las operaciones*. Nuestras estructuras mejoran la suma de costes del total de operaciones (en algunos casos no en cuanto al orden de complejidad, pero sí teniendo en cuenta las constantes que aparecen en la expresión de la función) respecto a las estructuras de datos solución definidas hasta el momento, y en algunos casos exigen un menor número de variables de programa.

- Definimos un modelo basado en la utilización de grafos para definir soluciones al *problema de los rangos variables*. Definimos grafos dentro de este modelo, que representan soluciones equivalentes a las definidas en el modelo matricial. Los algoritmos matriciales utilizados en la construcción de nuestras matrices solución, encuentran su traducción en las operaciones de adición de nodos y arcos que se ejecutan en el proceso de construcción de nuestros grafos solución.

- El análisis de la suma de costes de las operaciones que corresponde a las estructuras definidas en [19] no es para nada trivial, no se deduce fácilmente de la propia definición de las estructuras. Hemos realizado dichos cálculos en la parte final de este capítulo. El análisis resulta especialmente complicado en el caso de los árboles solución tratados en el apartado 5.4.1. A cambio, creemos que es un estudio que puede tener interés por si mismo.

Comenzamos definiendo nuestro modelo matricial para el estudio del *problema de los rangos variables*.

5.1 Modelo Matricial para el Problema de los Rangos Variables

Nos inspiramos en el modelo algebraico definido por M.L.Fredman (ver [14] o apartado 2.3.1 del Capítulo 2) para el estudio del *problema de las sumas parciales*. Recordamos que consiste en utilizar parejas de matrices de números enteros

R, U tales que $R \times U = T$, para definir las operaciones *Retrieve* y *Update* respectivamente. En particular, la fila *i-ésima* de la matriz R define la operación $Retrieve(i)$, y la columna *j-ésima* de la matriz U define las operaciónes $Update(j,x)$, $x \in S$.

El modelo matricial que proponemos para el estudio del *problema de los rangos variables*, incluye todos los programas que verifican:

- La estructura de datos solución es un vector $Z = (z_1, z_2 \ldots z_m)$ que toma valores en el semigrupo S.
- La operación $Retrieve(i,j)$ se ejecuta sumando un subconjunto de variables de Z.
- La operación $Update(j,x)$ se ejecuta incrementando un subconjunto de las variables de Z en cantidades que dependen linealmente de x.

Nuestro modelo matricial consiste en triplas $< Z, R, U >$ donde $Z = \{z_1 \ldots z_m\}$ es un conjunto de variables que toman valores en S, $R = (r_{i,j})$ es una matriz de ceros y unos de dimensión $\frac{n(n+1)}{2} \times m$, y $U = (u_{i,j})$ es una matriz de ceros y unos de dimensión $m \times n$ (m, el número de variables utilizadas, puede variar).

Asociados a una tripla $< Z, R, U >$, se definen los programas

Definición 7

Dada una tripla $< Z, R, U >$, *dentro del modelo matricial para el problema de los rangos variables de tamaño* n, *con* $Z = \{z_1 \ldots z_m\}$, *definimos los siguientes algoritmos para la implementación de las operaciones Update y Retrieve:*

- $Update(j,x)$: *for l:=1 to m do* $[z_l \leftarrow z_l + u_{l,j}\ x]$
- $Retrieve(i,j)$: *output* $\sum_{l=1}^{m} r_{k,l} z_l$, *siendo* $k = \sum_{s=0}^{i-2}(n-s) + (j - i + 1)$

El siguiente lema establece una condición sobre R, U, que implica la corrección de los programas que acabamos de definir.

Lema 7

Sea H la matriz de dimensión $[\frac{n(n+1)}{2}] \times n$ definida por:

$$H_{i,j} = \begin{cases} 1 & si \quad l \leq j \leq l + (i - w_{l-1} - 1) \\ 0 & e.o.c \end{cases}$$

donde

$$w_k = \sum_{l=0}^{k-1} (n - l) \ , \quad k = 0 \ldots n \ ,$$

$$i \in \{w_{l-1} + 1 \ldots w_l\} \ , \quad l = 1 \ldots n.$$

Entonces los programas dados en la Definición 7 son correctos si y sólo si $R \times U = H$.

Demostración

Al igual que en otras demostraciones similares realizadas en esta misma tesis, nos inspiramos en el modelo de demostración utilizado en [14] (ver *Lema 1* de [14] o consultar el apartado 2.3.1 del Capítulo 2). Tenemos que considerar el efecto de ejecutar consecutivamente las operaciones

Retrieve(i,j), Update(r,x), Retrieve(i,j)

para cualesquiera $i, j, r \in \{1 \ldots n\}$, x elemento del semigrupo conmutativo cuyos valores se desea almacenar.

Sean q_1, q_2 los output producidos como resultado de ejecutar las dos operaciones Retrieve(i,j), la primera y la segunda respectivamente.

A partir de la definición de las operaciones dada en el planteamiento general del problema de los rangos variables (ver el apartado 2.3.2 del Capítulo 2), es obvio que los programas dados en la Definición 7 representan una solución correcta si y sólo si

$$q_2 - q_1 = \begin{cases} x & i \leq r \leq j \\ 0 & \text{e.o.c.} \end{cases}$$

Pero

$$q_1 = \sum_{l=1}^{m} r_{k,l} z_l \ , \quad q_2 = \sum_{l=1}^{m} r_{k,l} \left(z_l + u_{l,r} x\right) \quad \text{con} \quad k = \sum_{s=0}^{i-2} (n-s) + (j-i+1) = w_{i-1} + (j-i+1),$$

y por tanto

$$q_2 - q_1 = \sum_{l=1}^{m} (r_{k,l} u_{l,r})\, x.$$

Aplicando ahora la definición de la matriz H y dado que

$$\left[k \;=\; \sum_{s=0}^{i-2} (n-s) + (j-i+1) \right] \;\Longrightarrow\; k \in \{ w_{i-1} + 1 \ldots w_i \}\ ,$$

tenemos

$$H_{k,r} = \begin{cases} 1 & \text{si} \quad i \;\leq\; r \;\leq\; i + (k - w_{i-1} - 1) \\ 0 & \text{e.o.c} \end{cases}$$

Pero $i + (k - w_{i-1} - 1) = i + (j - i + 1) - 1 = j$, y el lema se deduce inmediatamente ■

Observación 4

Observese que si T^n es la matriz triangular de dimensión $n \times n$ consistente en ceros por encima de la diagonal principal y unos en y por debajo de la diagonal principal - es decir, la matriz asociada al problema de las sumas parciales de tamaño n - se verifica que para todo $k = 0 \ldots (n-1)$,

$$H_{[w_k+1 \ldots w_{k+1}],[k+1 \ldots n]} = T^{n-k}$$

$$H_{[w_k+1 \ldots w_{k+1}],[1 \ldots k]} = 0$$

siendo $w_i = \sum_{s=0}^{i-1} (n-s)$ para $i = 0 \ldots n$, y $H_{[i \ldots j],[r \ldots s]}$ la caja de la matriz H formada por las filas del intervalo $[i \ldots j]$ y las columnas del intervalo $[r \ldots s]$.

Por tanto, para cualquier tamaño de problema n, podemos expresar la matriz correspondiente H^n en función de la matriz T de distintas dimensiones como:

$$H^n = \begin{pmatrix} T^n \\ \vec{T}^{n-1} \\ \vec{T}^{n-2} \\ \vdots \\ \vec{T}^{n-(n-1)} \end{pmatrix}$$

siendo

$$\vec{T}^{n-i} = \left(\begin{array}{cccc|c} 0 & \dots & \overset{i)}{\dots} & 0 & \\ \vdots & & & \vdots & \\ \overset{n-i)}{} & & & & T^{n-i} \\ \vdots & & & \vdots & \\ 0 & \dots & \dots & 0 & \end{array}\right) \qquad i = 1 \dots (n-1),$$

Ejemplo 1

A continuación mostramos la matriz H para el problema de los rangos variables de tamaños $n = 2$ y $n = 4$, a las que denominamos H^2 y H^4 respectivamente. Hemos trazado líneas que resaltan la división lógica por cajas.

$$H^2 = \left(\begin{array}{cc} 1 & 0 \\ 1 & 1 \\ - & - \\ 0 & 1 \end{array}\right), \qquad H^4 = \left(\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ - & - & - & - \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ - & - & - & - \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ - & - & - & - \\ 0 & 0 & 0 & 1 \end{array}\right)$$

Podemos proceder ahora a probar un resultado que impone alguna restricción sobre el número de variables $z_1 \dots zm$ empleadas para imlementar las soluciones al *problema de los rangos variables.*

Lema 8

Sean R, U dos matrices de ceros y unos de dimensiones respectivas $n \times m$ y $m \times n$, y tales que $R \times U = H^n$.

Entonces necesariamente $m \geq n$.

Demostración

$R \times U = H^n \Rightarrow R_{[1\ldots n],[1\ldots m]} \times U = T^n$, siendo $R_{[1\ldots n],[1\ldots m]}$ la caja de la matriz R formada por las n primeras filas. Tal y como se demostró en el Lema 1 del Capítulo 3 de esta tesis, eso implica que $m \geq n$ ■

La siguiente observación, establece la correspondencia entre las soluciones al *problema de los rangos variables* dentro de nuestro *modelo matricial* y dentro del marco establecido en [19] y expuesto en el apartado 2.3.2 del Capítulo 2.

Observación 5

Dada una tripla $< Z, R, U >$ *solución para el problema de los rangos variables de dimensión n dentro del modelo matricial, con*

$$Z = (z_1 \ldots z_m) \quad R = (r_{i,j})_{i=1\ldots\frac{n(n+1)}{2},j=1\ldots m} \quad U = (u_{i,j})_{i=1\ldots m,j=1\ldots n}$$

y dados $\{Y_1 \ldots Y_m\}$, $\{R_{i,j} \ / \ i,j = 1\ldots n\}$ *definidos como en [19] (ver apartado 2.3.2 en el Capítulo 2), se tiene que*

$$r_{i,j} = \begin{cases} 1 & \textit{si y sólo si} \quad j \in R_{kl}, \quad \textit{siendo} \quad i = (l-k+1) + \sum_{s=0}^{k-2}(n-s) \\ 0 & \textit{e.o.c} \end{cases}$$

$$u_{i,j} = \begin{cases} 1 & \textit{si y sólo si} \quad j \in Y_i \\ 0 & \textit{e.o.c} \end{cases}$$

A continuación definimos la complejidad asociada a las operaciones dentro del modelo matricial.

Definición 8

Dada una tripla $< Z, R, U >$ *solución para el problema de los rangos variables de dimensión n dentro del modelo matricial, con* $Z = (z_1 \ldots z_m)$, $R = (r_{i,j})_{i=1\ldots\frac{n(n+1)}{2},j=1\ldots m}$, $U = (u_{i,j})_{i=1\ldots m,j=1\ldots n}$, *definimos*

- *Complejidad asociada a la operación* $Retrieve(i,j)$:

$$\#\{r_{k,l} \ / \ r_{k,l} \neq 0 \ \wedge \ (1 \leq l \leq m)\} \quad \textit{siendo} \quad k = (j-i+1) + \sum_{s=0}^{i-2}(n-s).$$

- *Complejidad asociada a la operación* $Update(j,x)$:

$$\#\{u_{l,j} \ / \ u_{l,j} \neq 0 \ \wedge \ (1 \leq l \leq m)\}$$

5.1.1 Cálculo de una Cota Inferior para la Suma de las Complejidades Medias

En esta sección, presentamos un procedimiento para el cálculo de una cota inferior para la suma de la complejidad media de las operaciones *Retrieve* y la complejidad media de las operaciones *Update*. El método nos permite mejorar ligeramente la cota establecidas hasta el momento.

Trabajamos dentro del modelo matricial presentado en la sección anterior.

Sean R, U dos matrices de ceros y unos que representan una solución al *problema de los rangos variables* de tamaño n, o lo que es lo mismo, tales que $R \times U = H^n$.

Recalcamos que no se trata de establecer una cota inferior para la complejidad media de todas las operaciones en conjunto, lo que equivaldría a establecer el mínimo número de unos que deberían sumar entre las dos matrices y dividir esa cantidad por el número de operaciones posibles, $n + \frac{n(n+1)}{2}$, si no de considerar la complejidad media de las operaciones *Retrieve* por un lado, y la correspondiente a las operaciones *Update* por otro.

Sea m el número de columnas de R y de filas de U. La complejidad media de las operaciones *Update* viene dada por

$$\rho = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} u_{i,j}}{n} \ ,$$

y la complejidad media de las operaciones *Retrieve* por

$$t = \frac{\sum_{i=1}^{n(n+1)/2} \sum_{j=1}^{m} r_{i,j}}{n(n+1)/2} \ .$$

En [19] se establece que para cualquier estructura de datos solución del *problema de los rangos variables* de tamaño n, se tiene que $\rho + t = \Omega(logn)$ (ver *Teorema 1* de [19]).

Recordamos que una función f es $\Omega(g(n))$ si existen constantes c, n_0 tales que para todo $n \geq n_0$ se verifica $f(n) \geq cg(n)$, y por tanto el resultado establece una cota inferior para la suma de las complejidades medias.

Examinamos la demostración del `Teorema 1` de [19] para comprobar el grado de precisión al que se ha llegado al establecer la cota inferior, y vemos que, más concretamente, el resultado obtenido es

$$(2n^2/9)\log_e n + O(n^2) \quad \leq \quad 2n^2\rho + 2\frac{n(n+1)}{2}t \ .$$

Aquí proporcionamos un procedimiento distinto que, de forma sencilla, nos permite obtener una cota inferior para $2n^2\rho + n(n+1)t$ basandonos en el conocimiento que ya tenemos de una cota inferior optimal para la complejidad media (conjunta) de las operaciones *Update* y *Retrieve* en relación al *problema de las sumas parciales* de tamaño n. Recordamos que el *modelo algebraico* de M.L.Fredman (ver apartado 2.3.1 del Capítulo 2 o consultar [14]) para el estudio de las cotas inferiores de dicho problema, consiste en parejas de matrices de ceros y unos cuyo producto es la matriz T^n - matriz triangular de dimensión $n \times n$ consistente en ceros por encima de la diagonal principal y unos en y por debajo de la diagonal principal.

Como ya hemos visto en el Capítulo 3 de esta tesis (ver Teorema 2), se demuestra que para $2^k \leq n < 2^{k+1}$, se tiene que si $A \times B = T^n$, siendo A y B matrices de dimensión $n \times m$ y $m \times n$ respectivamente, entonces

$$\begin{aligned} \varphi(n,m) &= \sum_{i=1}^{n}\sum_{j=1}^{m} a_{i,j} + \sum_{i=1}^{m}\sum_{j=1}^{n} b_{i,j} \\ &\geq n[\log_2 n] + [\log_2 n] + (m - 2^k) + 2 \end{aligned}$$

A continuación establecemos nuestra cota inferior para $[2n^2\rho + n(n+1)t]$. Se debe observar que el termino que *manda* en la expresión que obtendremos es

$$n^2(\log_2 n - \frac{1}{2}\log_2 e)$$

mientras que en el resultado obtenido en [19] es

$$(2n^2/9)\log_e n.$$

Teorema 5 *Sean R, U dos matrices de ceros y unos tales que $R \times U = H^n$.*

Entonces se tiene que

$$2n^2\rho + n(n+1)t \;\geq\; n^2(\log_2 n - \frac{1}{2}\log_2 e) + 2n\log_2 n + 2n + \frac{10}{4}\log_2 e$$

siendo

$$\rho = \frac{\sum_{j=1}^{n}\sum_{i=1}^{m} u_{i,j}}{n}\ , \qquad t = \frac{\sum_{i=1}^{n(n+1)/2}\sum_{j=1}^{m} r_{i,j}}{n(n+1)/2}$$

y m el número de columnas de R y filas de U.

Demostración

Tenemos que

$$R \times U = H^n = \begin{pmatrix} T^n \\ \vec{T}^{n-1} \\ \vec{T}^{n-2} \\ \vdots \\ \vec{T}^{n-(n-1)} \end{pmatrix}$$

donde

$$\vec{T}^{n-i} = \left(\begin{array}{cccc|c} 0 & \dots & \overset{i)}{\dots} & 0 & \\ \vdots & & & \vdots & \\ \overset{n-i)}{} & & & & T^{n-i} \\ \vdots & & & \vdots & \\ 0 & \dots & \dots & 0 & \end{array}\right) \qquad i = 1 \dots (n-1),$$

Denominamos $R_{[i\dots j],[1\dots m]}$ al bloque formado por las filas $[i \dots j]$ de R, con $1 \leq i \leq j \leq \frac{n(n+1)}{2}$.

Consideramos la matriz R dividida en n bloques, $R_1 \dots R_n$, el primero de ellos consistente en las primeras n filas, el segundo formado por las siguientes $(n-1)$ filas $\dots$, así hasta llegar a un último bloque, R_n, formado únicamente por la última fila de R.

Dicho más formalmente, llamando $z_i = \sum_{j=0}^{i-1}(n-j)$, $i = 0 \dots n$, tenemos $R_i = R_{[z_{i-1}+1\dots z_i],[1\dots m]}$.

Usando esta notación, tenemos que

$$R_i \times U \quad = \quad T^{n-i+1}$$

Aplicamos el resultado conocido para el *problema de las sumas parciales* a cada pareja R_i, U con $i = 1 \ldots n$ y tenemos que

$$\sum_{l=z_{i-1}+1}^{z_i} \sum_{s=1}^{m} r_{l,s} + \sum_{l=1}^{m} \sum_{s=1}^{n} u_{l,s} \quad \geq \quad (n-i+1)[\log_2(n-i+1)] + [\log_2(n-i+1)] + (m-2^k) + 2$$

Por tanto, y desestimando el factor $(m - 2^k)$

$$\begin{aligned} \sum_{i=1}^{n(n+1)/2} \sum_{j=1}^{m} r_{i,j} + n \left[\sum_{i=1}^{m} \sum_{j=1}^{n} u_{i,j} \right] \quad &\geq \quad [(n+1)\log_2 n + 2] + [n\log_2(n-1) + 2] + \ldots \\ &\quad \ldots + [(2+1)\log_2 2 + 2] + 2 \\ &= \quad [(n+1)\log_2 n + n\log_2(n-1) + (2+1)\log_2 2] + 2n \\ &= \quad \delta(n) + 2n \end{aligned}$$

Antes de continuar aproximando el valor de lo que hemos llamado $\delta(n)$ con la mayor precisión posible, observamos que

$$\sum_{i=1}^{n(n+1)/2} \sum_{j=1}^{m} r_{i,j} + n \left[\sum_{i=1}^{m} \sum_{j=1}^{n} u_{i,j} \right] \quad = \quad \frac{n(n+1)}{2} t + n^2 \rho \ ,$$

y recordamos que queremos establecer una cota inferior para $2n^2\rho + n(n+1)t$.

Llamamos $f(x) = (x+1)\log_2 x$.

Tenemos que

$$\int_1^n f(x)dx \leq \delta(n) \leq \int_2^{n+1} f(x)dx$$

Teniendo en cuenta que $\log_2 x = \log_2 e \log_e x$, integramos la expresión

$$(\log_2 e) \int (x+1)\log_e x \ dx$$

Integramos por partes,

$$\begin{aligned} u = \log_e x \quad &\Rightarrow \quad du = \frac{dx}{x} \\ dv = (x+1)dx \quad &\Rightarrow \quad v = \frac{(x+1)^2}{2} \end{aligned}$$

y tenemos (el factor constante $\log_2 e$ será tenido en cuenta al final)

$$\begin{aligned}\int (x+1)\log_e x dx &= \frac{(x+1)^2}{2}\log_e x - \int \frac{x^2+2x+1}{2x}dx \\ &= \frac{(x+1)^2}{2}\log_e x - \frac{1}{2}\int x dx - \int dx - \frac{1}{2}\int \frac{dx}{x} \\ &= \frac{(x+1)^2}{2}\log_e x - \frac{1}{4}x^2 - x - \frac{1}{2}\log_e x\end{aligned}$$

Por tanto

$$\int_1^n f(x)dx = \frac{(n+1)^2}{2}\log_e n - \frac{1}{4}n^2 + \frac{5}{4} - n - \frac{1}{2}\log_e n$$

Tenemos entonces

$$\begin{aligned}n^2\rho + \frac{n(n+1)}{2}t &\geq \delta(n) + 2n \\ &\geq 2n + (\log_2 e)\left[\frac{(n+1)^2}{2}\log_e n - \frac{1}{4}n^2 + \frac{5}{4} - n - \frac{1}{2}\log_e n\right] \\ &= 2n + (\log_2 e)\left[\frac{n^2}{2}\log_e n + n\log_e n - n - \frac{1}{4}n^2 + \frac{5}{4}\right] \\ &\geq \frac{n^2}{2}\log_2 n + n\log_2 n + n - \frac{1}{4}n^2(\log_2 e) + \frac{5}{4}(\log_2 e)\end{aligned}$$

De aquí se obtiene, multiplicando por dos ambos miembros de la desigualdad, el resultado que establece el teorema,

$$2n^2\rho + n(n+1)t \geq n^2\log_2 n + 2n\log_2 n + 2n - \frac{1}{2}n^2(\log_2 e) + \frac{10}{4}(\log_2 e) \blacksquare$$

Observación 6

Obsérvese que del resultado del teorema anterior se deduce efectivamente una cota $\Omega(\log n)$ *para* $\rho + t$: *dividimos ambos términos de la desigualdad final por* $2n^2$ *y obtenemos* $\rho + \left(\frac{1}{2} + \frac{1}{2n}\right) t \geq \log_4 n + \frac{\log_2 n}{n} + \frac{1}{n} - \frac{1}{4}(\log_2 e) + \frac{10}{4}(\frac{\log_2 e}{2n^2})$, *y por tanto, dado que* $\frac{1}{2} + \frac{1}{2n} \leq 1 \quad \forall n \geq 1$, *se concluye que* $\rho + t \in \Omega(\log n)$.

5.1.2 Matrices Solución para el Problema de los Rangos Variables

El objetivo último de esta sección es definir parejas de matrices de ceros y unos cuyo producto es la matriz H - es decir, representan soluciones al *problema de los rangos variables* - intentando minimizar la suma del número total de unos presentes en ambas matrices, y por tanto la complejidad media de las operaciones. Daremos una definición recursiva en el apartado 5.1.3 de esta sección. En particular definiremos matrices $R = (r_{i,j})$, $U = (u_{i,j})$ cuyo producto es H^n para cualquier dimensión n de la forma 2^k con $k \in \mathbf{N}^+$.

Recordamos que la complejidad media se calcula dividiendo el número total de unos por el número de operaciones distintas. Así, si tratamos con el problema de tamaño n y llamamos

$$\psi(n) = \sum_{i=1}^{\frac{n(n+1)}{2}} \sum_{j=1}^{m} r_{i,j} + \sum_{i=1}^{m} \sum_{j=1}^{n} u_{i,j},$$

siendo m el número de variables $z_1 \ldots z_m$ utilizadas para implementar la solución (m puede variar, aunque ha de ser necesariamente mayor o igual que n, como se estableció en el Lema 8 de este capítulo), entonces la complejidad media viene dada por

$$\frac{\psi(n)}{n + \frac{n(n+1)}{2}}.$$

(n es el número de operaciones $Update(j, x)$ distintas en función del primer argumento, y $\frac{n(n+1)}{2}$ es el número de argumentos distintos posibles para una operación $Retrieve(i, j)$).

Probaremos que nuestras matrices verifican

$$\psi(n) = \frac{3}{2}n^2 - \frac{3}{2}n \log n + \frac{9}{2}n - 2 \log n - 4$$

lo que implica una complejidad media cuyo orden de complejidad es constante. Nuestras matrices mejoran la suma de costes del total de operaciones correspondiente a las estructuras de datos solución definidas en [19].

Si llamamos $Var(n)$ al número de variables que requiere nuestra solución para el problema de tamaño n, desmostraremos también que

$$Var(n) \;=\; n\log_2 n - 2n + 2\log_2 n + 2.$$

Nuestras matrices solución son *buenas* en cuanto a la complejidad media de las operaciones consideradas en su conjunto, pero si se considera por un lado la complejidad media de las operaciones *Retrieve* y por otro la correspondiente a las operaciones *Update*, las primeras tienen complejidad constante mientras que el promedio para las segundas es de $\frac{n}{2} - \frac{\log n}{2}$. Evidentemente, el peso de las operaciones *Retrieve* en la media es mayor, puesto que hay $\frac{n(n+1)}{2}$ operaciones *Retrieve* distintas y sólo n operaciones *Update*. *El interés principal de nuestras estructuras reside en aplicaciones en las que todas las operaciones son equiprobables.*

En el proceso de construcción de las parejas de matrices, será necesaria la ejecución de algoritmos que a partir de una pareja inicial de matrices reducen el número de unos total presente en ambas sin afectar al producto. Tales algoritmos son aplicables a parejas de matrices distintas de las que nosotros vamos a construir.

Antes de proceder a demostrar formalmente los resultados mencionados, necesitamos introducir algo de notación y recordar algunos conceptos.

Recordamos que utilizamos un superíndice para especificar el tamaño del problema al que corresponde la matriz H. Así, H^n denota la matriz correspondiente al *problema de los rangos variables* de tamaño n, aunque la dimensión de H^n es $[\frac{n(n+1)}{2} \times n]$.

En caso de que hagamos referencia a la matriz T, solución del *problema de las sumas parciales*, coincide la dimensión del problema con la dimensión de la matriz, que como sabemos, es cuadrada. Así, T^n denota la matriz correspondiente al problema de las sumas parciales de tamaño n.

Llamaremos $I^m_{i,j}$ a la matriz resultante de permutar las filas *i-ésima* y *j-ésima* en la matriz identidad de dimensión $m \times m$, a la que denotamos por I^m . Para cualquier matriz M de dimensión $m \times n$, $I^m_{i,j} \times M$ devuelve la matriz M con las filas i, j intercambiadas de posición.

En general, si llamamos $I^m_\sigma = I^m_{i_1,j_1} \times I^m_{i_2,j_2} \times \ldots \times I^m_{i_k,j_k}$, el efecto de la multiplicación $I^m_\sigma \times M$ es intercambiar de posición las filas i_k y j_k de M,

después hacer lo mismo con las filas i_{k-1} y j_{k-1}, *después* con las filas i_{k-2} y $j_{k-2}\ldots$ hasta finalmente intercambiar las filas i_1, j_1.

Para cualquier $n = 2^k$, podemos conseguir cualquier reordenación por filas de la matriz H^n asociada al problema de los rangos variables de tamaño n , multiplicándola por una cierta transformada de la identidad I_σ, donde $\sigma \in Permutaciones\left(\binom{n+1}{2}\right)$, siendo $Permutaciones(k) = \{f : \{1\ldots k\} \longrightarrow \{1\ldots k\} \ / \ f \text{ biyectiva}\}$, $k \in \mathbf{N}^+$. Esto se debe a un resultado algebraico conocido, que afirma que cualquier permutación perteneciente a $Permutaciones(k)$ se puede expresar como composición de un cierto número de permutaciones de ese conjunto, tales que dejan fijos todos los elementos de $\{1\ldots k\}$ salvo dos de ellos.

El objetivo es que, finalmente, la matriz H^n quede reordenada como la matriz S^n cuya definición damos a continuación:

Definición 9 *Para todo* n *de la forma* 2^l *con* $l \in \mathbf{N}^+$ *definimos*

$$S^n = \begin{pmatrix} H^{\frac{n}{2}} & 0 \\ M_1^{\frac{n}{2}} & T^{\frac{n}{2}} \\ M_2^{\frac{n}{2}} & T^{\frac{n}{2}} \\ \vdots & \vdots \\ M_{\frac{n}{2}}^{\frac{n}{2}} & T^{\frac{n}{2}} \\ 0 & H^{\frac{n}{2}} \end{pmatrix}$$

donde, para cualquier $m \in \mathbf{N}^+$, $k = 1\ldots m$, *las matrices* M_k^m *son matrices cuadradas de dimensión* m *definidas por*

$$(M_k^m)_{i,j} = \begin{cases} 1 & si \quad k \leq j \\ 0 & e.o.c \end{cases}$$

En adelante, llamaremos $I_{H^n \to S^n}$ a la matriz I_σ que provoca la transformación de H^n en S^n. Es decir, tendremos que

$$I_{H^n \to S^n} \times H^n = S^n.$$

Naturalmente, para poder concluir la existencia de la matriz $I_{H^n \to S^n}$, tendremos que probar primero que el conjunto de filas de la matriz H^n es exactamente igual al conjunto de filasde la matriz S^n. Lo haremos en el Lema 10 .

Vemos un par de ejemplos sencillos en los que es fácil deducir la expresión concreta de la matriz $I_{H^n \to S^n}$.

Ejemplo 2 *En el caso* $n = 2$, *tenemos que*

$$H^2 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} = S^2$$

y por tanto $I_{H^2 \to S^2} = I^2$.

Vemos también el ejemplo de la transformación de H^4 *en la* S^4, *en el que se muestran las operaciones que es necesario realizar y la expresión que corresponde a la matriz* $I_{H^4 \to S^4}$.

Tenemos que

$$H^4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{I_{3,5}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{I_{4,5}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} = S^4$$

Por tanto,

$$I_{H^4 \to S^4} \;=\; I_\sigma \;=\; I_{4,5} \times I_{3,5}$$

siendo

$$\sigma : \{1 \ldots 10\} \longrightarrow \{1 \ldots 10\}, \qquad \sigma(3) = 4, \quad \sigma(4) = 5, \quad \sigma(5) = 3 \; \blacksquare$$

Observación 7

¿Cómo afecta este tipo de transformaciones *al planteamiento matricial del problema de los rangos variables? Sabemos que se trata de estudiar parejas de matrices de números enteros* R^n, U^n *tales que* $R^n \times U^n = H^n$. *Pero si tenemos que* $R^n \times U^n = H^n$, *entonces* $I_{H^n \to Sn} \times R^n \times U^n = S^n$, *así que podemos dar una reformulación equivalente del problema en la que se trata de estudiar parejas de matrices cuyo producto es la matriz* S^n. *En este caso debemos modificar el algoritmo que implementa las operaciones Retrieve dado en la Definición 7, de forma que ahora la definición de los programas asociados a una tripla* $< Z, R, U >$ *es la siguiente:*

Definición 10

Dada una tripla $< Z, R, U >$, *con* $Z = \{z_1 \dots z_m\}$, R, U *matrices de dimensión* $n \times m$ *y* $m \times n$ *respectivamente, definimos los siguientes algoritmos para la implementación de las operaciones Update y Retrieve:*

1. *Update*(j, x) : *for l:=1 to m do* $[z_l \leftarrow z_l + u_{l,j}\, x]$
2. *Retrieve*(i, j) : *output* $\sum_{l=1}^{m} r_{k,l} z_l$, *donde k viene dada por :*
   - *(a)* $k = \sum_{s=0}^{i-2} (\frac{n}{2} - s) + (j - i + 1)$ *si* $1 \leq i \leq j \leq \frac{n}{2}$
   - *(b)* $k = \sum_{s=0}^{i-2} (n - s) + (j - i + 1)$ *si* $\frac{n}{2} < i \leq j \leq n$
   - *(c)* $k = \frac{n}{2}(i-1) + (j - \frac{n}{2}) + \frac{\frac{n}{2}(\frac{n}{2}+1)}{2}$ *si* $i \leq \frac{n}{2}$, $j \geq \frac{n}{2}$

La idea intuitiva es que ahora cambia el número de fila (k) de la matriz R^n *asociado a una operación* $Retrieve(i, j)$, *pues algunas filas han modificado su posición.*

A efectos del estudio de la complejidad de las operaciones, el cambio de planteamiento no tiene ninguna incidencia, pues recordamos que el efecto de multiplicar una matriz cualquiera por una cierta I_σ *no altera el número de elementos no nulos de la matriz, sino que tan sólo intercambia de posición ciertas filas.*

El Teorema 9 a continuación, establece la condición sobre una pareja de matrices R, U, que implica la corrección de los programas de la definición 10 que acabamos de exponer.

Lema 9

Sea S^n *la matriz de dimensión* $[\frac{n(n+1)}{2}] \times n$ *definida como en la Definición 9.*

Entonces los programas de la definición 10 son correctos si y sólo si $R \times U = S^n$.

Demostración

La demostración es trivial tomando como modelo de demostración la realizada en el Lema 7 e introduciendo los cambios que correspondan en función de la nueva definición del algoritmo que implementa las operaciónes *Retrieve* y de la forma particular de la matriz S^n ■

Tenemos pendiente la demostración de que para cualquier dimensión n de la forma 2^k, $k \in \mathbf{N}^+$, el conjunto de filas de H^n es igual al conjunto de filas de S^n. Lo probamos a continuación, en el Lema 10 .

Lema 10

Sean las matrices H^n, S^n *definidas como en 7 y 9 respectivamente, con* $n = 2^k$, $k \in \mathbf{N}^+$. *Entonces se verifica que cada una de ellas es una reordenación por filas de la otra.*

Demostración

A partir de las definiciones dadas para las matrices H^n, T^n, M_i^n tenemos

$$H^n = \begin{pmatrix} T^n \\ \vec{T}^{n-1} \\ \vec{T}^{n-2} \\ \vdots \\ \vec{T}^{n-(n-1)} \end{pmatrix} \qquad Obs.(1)$$

siendo

$$\vec{T}^{n-i} = \left(\begin{array}{cccc|c} 0 & \dots & \overset{i)}{\dots} & 0 & \\ \vdots & & & \vdots & \\ \overset{n-i)}{} & & & & T^{n-i} \\ \vdots & & & \vdots & \\ 0 & \dots & \dots & 0 & \end{array}\right) \qquad i = 1 \dots (n-1),$$

y podemos descomponer matrices de la siguiente manera

$$\vec{T}^{n-i} = \begin{pmatrix} \vec{T}^{\frac{n}{2}-i} & 0 \\ M_i^{\frac{n}{2}} & T^{\frac{n}{2}} \end{pmatrix} \qquad i = 1 \dots (\frac{n}{2} - 1),$$

$$T^n = \begin{pmatrix} T^{\frac{n}{2}} & 0 \\ M_1^{\frac{n}{2}} & T^{\frac{n}{2}} \end{pmatrix}$$

Observamos que descomponiendo H^n como en (1), se aprecia que

$$\begin{pmatrix} \vec{T}^{n-\frac{n}{2}} \\ \vec{T}^{n-(\frac{n}{2}+1)} \\ \vec{T}^{n-(\frac{n}{2}+2)} \\ \vdots \\ \vec{T}^{n-(n-1)} \end{pmatrix} = \left(\begin{array}{cccc|c} 0 & \dots & \overset{\frac{n}{2})}{\dots} & 0 & \\ \vdots & & & \vdots & \\ \overset{\frac{n^2+2n}{8})}{} & & & & H^{\frac{n}{2}} \\ \vdots & & & \vdots & \\ 0 & \dots & \dots & 0 & \end{array}\right)$$

(obsérvese que $\frac{n^2+2n}{8} = \frac{\frac{n}{2}(\frac{n}{2}+1)}{2}$, es el número de filas de la matriz $H^{\frac{n}{2}}$, cuyo número de columnas es $\frac{n}{2}$).

Por tanto en el proceso de reordenación de H^n para obtener la matriz

$$S^n = \begin{pmatrix} H^{\frac{n}{2}} & 0 \\ M_1^{\frac{n}{2}} & T^{\frac{n}{2}} \\ M_2^{\frac{n}{2}} & T^{\frac{n}{2}} \\ \vdots & \vdots \\ M_{\frac{n}{2}}^{\frac{n}{2}} & T^{\frac{n}{2}} \\ 0 & H^{\frac{n}{2}} \end{pmatrix}$$

no es necesario aplicar ninguna transformación a las últimas $\frac{n^2+2n}{8}$ filas de H^n. $Obs.(2)$

Por otra parte, nos fijamos en los bloques formados por las $\frac{n}{2}$, $(\frac{n}{2}-1)$, $(\frac{n}{2}-2)\ldots 2,1$ primeras filas respectivamente - aparecen resaltados en negrita - de las descomposiciones matriciales

$$T^n = \begin{pmatrix} \mathbf{T^{\frac{n}{2}}} & \mathbf{0} \\ M_1^{\frac{n}{2}} & T^{\frac{n}{2}} \end{pmatrix}, \qquad \vec{T}^{n-i} = \begin{pmatrix} \mathbf{\vec{T}^{\frac{n}{2}-i}} & \mathbf{0} \\ M_i^{\frac{n}{2}} & T^{\frac{n}{2}} \end{pmatrix} \quad \mathbf{i = 1 \ldots (\frac{n}{2} - 1)}.$$

Con esos $\frac{n}{2}$ bloques podemos formar

$$\begin{pmatrix} T^{\frac{n}{2}} & 0 \\ \vec{T}^{\frac{n}{2}-1)} & 0 \\ \vec{T}^{\frac{n}{2}-2)} & 0 \\ \vdots & \vdots \\ \vec{T}^{\frac{n}{2}-(\frac{n}{2}-1)} & 0 \end{pmatrix} = \left(\begin{array}{c|ccc} & 0 & \ldots \quad \ldots^{\frac{n}{2})} & 0 \\ H^{\frac{n}{2}} & \vdots & & \vdots \\ & {\scriptstyle \frac{n^2+2n}{8})} & & \\ & \vdots & & \vdots \\ & 0 & \ldots \quad \ldots & 0 \end{array} \right)$$

que constituye el primer bloque de $\frac{n^2+2n}{8}$ filas de la matriz S^n, esto es

$$\left[H^{\frac{n}{2}} \mid 0 \right] \qquad Obs.(3)$$

En cuanto a los $\frac{n}{2}$ bloques centrales de S^n provienen de los bloques formados por las $\frac{n}{2}$ últimas filas respectivamente - aparecen resaltados en negrita - de las descomposiciones matriciales

$$T^n = \begin{pmatrix} T^{\frac{n}{2}} & 0 \\ \mathbf{M_1^{\frac{n}{2}}} & \mathbf{T^{\frac{n}{2}}} \end{pmatrix}, \qquad \vec{T}^{n-i} = \begin{pmatrix} \vec{T}^{\frac{n}{2}-i} & 0 \\ \mathbf{M_i^{\frac{n}{2}}} & \mathbf{T^{\frac{n}{2}}} \end{pmatrix} \quad \mathbf{i = 1 \ldots (\frac{n}{2} - 1)},$$

formando efectivamente el bloque central de S^n.

$$\begin{bmatrix} M_1^{\frac{n}{2}} & T^{\frac{n}{2}} \\ M_2^{\frac{n}{2}} & T^{\frac{n}{2}} \\ \vdots & \vdots \\ M_{\frac{n}{2}}^{\frac{n}{2}} & T^{\frac{n}{2}} \end{bmatrix} \qquad Obs.(4)$$

El resultado que se quiere demostrar se deduce de las $Obs.(2), (3)$ y (4).

Podemos concluir trivialmente el siguiente corolario.

Corolario 4

Sean las matrices H^n, S^n *definidas como en el Lema 7 y la Definición 9 respectivamente, con* $n = 2^k$, $k \in \mathbf{N}^+$.

Existe una matriz cuadrada P *tal que* $P \times H^n = S^n$.

Demostración

El resultado es trivial a partir del lema anterior. Una vez demostrado que las matrices H^n, S^n contienen las mismas filas,concluímos que tan sólo hay que reordenarlas de la manera adecuada, algo que siempre es posible hacer multiplicando por la matriz I_σ correspondiente, matriz que en este caso hemos denominado $I_{H^n \to S^n}$.

5.1.3 Definicion Recursiva de Nuestras Matrices Solución

En este apartado daremos una definición recursiva de nuestras parejas de matrices R^n, U^n en función del tamaño n del problema. Las matrices verifican $R^n \times U^n = H^n$.

Como ya se ha mencionado, la definición es válida para valores de la forma $n = 2^k$.

Haremos referencia a *bloques* de filas consecutivas de la matriz R^n, a la que consideramos dividida en n bloques horizontales, el primero formado por las n primeras filas, el segundo por las $(n-1)$ siguientes, el tercero por las $(n-2)$ ... así hasta el bloque *(n-1)-ésimo* que consta de dos filas y el bloque *n-ésimo*

que consiste exclusivamente en la última fila. Denotamos por R_i^n al *i-ésimo* bloque de R^n , y por $R_i^n(j)$ a la fila *j-ésima* de dicho bloque, de forma que la matriz R^n viene dada por

$$R^n = \begin{pmatrix} R_1^n \\ R_2^n \\ \vdots \\ R_n^n \end{pmatrix}$$

(obsérvese que si la dimensión de R^n es $\frac{n(n+1)}{2} \times m$, entonces la dimensión de cada bloque R_i^n es $(n-i+1) \times m$).

La construcción de las parejas R^n, U^n requiere la aplicación de una función llamada *Refinamiento*. Podemos ver esta función como un proceso que consta de dos etapas: la primera de ellas consiste en la ejecución de una secuencia de `pasos de ampliacion`, concepto que expondremos en la Definición 12, y la segunda en efectuar una reordenación de R^n por filas mediante la multiplicación por una cierta matriz transformada de la identidad I_σ.

La Definición 11 y el Lema 11 a continuación son necesarios para definir el concepto de `paso de ampliacion`.

Definición 11

Dada una matriz M de ceros y unos, decimos que dos columnas i, j son disjuntas si el conjunto de filas $\{k \ / \ m_{k,i} = 1 = m_{k,j}\}$ es vacío. De igual forma, decimos que dos filas i, j de M son `disjuntas` *si el conjunto de columnas $\{k \ / \ m_{i,k} = 1 = m_{j,k}\}$ es vacío.*

Lema 11

Sean A, B dos matrices de ceros y unos tales que $A \times B = S^n$, de dimensiones respectivas $\frac{n(n+1)}{2} \times m$ y $m \times n$. Supongamos que existen dos columnas i, j de A que no son disjuntas, y sea $\{l_1 \ldots l_q\}$ el conjunto de filas de A para las cuales se verifica $a_{l_k,i} = 1 = a_{l_k,j}$, $k = 1 \ldots q$. Entonces las filas i, j de B son disjuntas.

Demostración

Supongamos que existiese una columna l_0 de B tal que $b_{i,l_0} = 1 = b_{j,l_0}$. En ese caso, se tendría que para cualquiera de los índices $k = 1 \ldots q$, el producto de la fila l_k de A por la columna l_0 de B sería mayor que 1 ■

A continuación definimos el concepto de `paso de ampliacion`.

Definición 12

Sean A, B *dos matrices de ceros y unos tales que* $A \times B = S^n$, *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$.

Supongamos que en A *tenemos un conjunto de columnas* $C = \{c_1 \ldots c_l\}$, $l \geq 2$, *para las cuales existe un conjunto no vacío* maximal - *incluye* todas *las filas que cumplen la siguiente condición - de filas* $F = \{f_1, \ldots, f_q\}$ *tales que*

$$[\ a_{f_i,c_j} = 1\ ] \qquad \forall c_j \in C, \quad \forall f_i \in F.$$

Definimos el `paso de ampliacion` *asociado a los conjuntos* C, F *como la ejecucion de las acciones siguientes sobre* A *y* B *:*

1. *Se inserta una nueva columna* z_0 *en* A *tal que*

$$[\ a_{i,z_0} = 1 \Leftrightarrow i \in F\ ] \qquad \forall\ i = 1 \ldots \frac{n(n+1)}{2}$$

2. *Se añade una nueva fila* z_0 *en* B *tal que*

$$[\ b_{z_0,j} = \sum\{\ b_{k,j}\ /\ k \in C\ \}\ ] \qquad \forall\ j = 1 \ldots m.$$

3. *Las columnas* $c_1 \ldots c_l$ *de* A *sufren la siguiente modificación:*

$$[\ a_{f_i,c_k} := 0\ ] \qquad \forall c_k \in C, \quad \forall f_i \in F$$

El siguiente lema prueba que si aplicamos un `paso de ampliacion` a una pareja de matrices cuyo producto es la matriz S^n, el producto de las matrices resultantes sigue siendo la misma matriz S^n.

Lema 12

Sean A, B *dos matrices de ceros y unos tales que* $A \times B = S^n$, *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$.

Si se ejecuta un `paso de ampliacion` *sobre ambas matrices, el producto de las matrices resultantes continúa siendo la matriz* S^n.

Demostración

Sea $C = \{c_1 \dots c_l\}$, $l \geq 2$, un conjunto de columnas de A y sea $F = \{f_1 \dots f_q\}$ un conjunto maximal de filas de A tales que

$$[\ a_{f_i,c_j} = 1\ ] \qquad \forall\ c_j \in C, \quad \forall\ f_i \in F.$$

Sea A', B' la pareja de matrices resultante de aplicar el `paso de ampliacion` asociado a C y F sobre A, B, y sea z_0 el índice que señala la posición en que se inserta una nueva columna en A y una nueva fila en B.

Dadas una fila f de A' y una columna c de B', distinguimos los siguientes casos:

1. $f \in F$:

   (a) el producto era 0 antes del `paso de ampliacion`: en este caso aparece un nuevo 1 en la posición a'_{f,z_0}, pero $b'_{z_0,c}$ es trivialmente 0 por ser cero el producton antes del paso de ampliación y por el punto 2 de la Definición 12.

   (b) el producto era 1 antes del `paso de ampliacion`: si el índice l_0 para el cual $a_{f,l_0} = 1 = b_{l_0,c}$ pertenece al conjunto C, el producto seguirá siendo 1 tras el `paso de ampliacion`, pues por la Definición 12 se tendrá que $a'_{f,z_0} = b'_{z_0,c} = 1$ y $a'_{f,l_0} = 0$.
   Si por el contrario $l_0 \notin C$, el producto no se ve afectado por el `paso de ampliacion`.

2. $f \notin F$: la única modificación que sufre la fila f es que aparece un nuevo 0 en la posición a'_{f,z_0}, por lo que el producto no varía fuera cual fuese.

Ahora ya estamos en condición de poder dar una definición recursiva de nuestras parejas de matrices.

Definición 13

Definimos recursivamente parejas de matrices R^n, U^n *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$, *con* n *de la forma* 2^k, $k \in \mathbf{N}^+$, *de la siguiente manera:*

1. $n = 2 : \ < R^2, U^2 > = < H^2, I^2 >$

2. $n = 2^k : \ < R^{2n}, U^{2n} > =$ *Refinamiento* ($\widehat{R^{2n}}, \widehat{U^{2n}}$), *siendo:*

(a) $$\widehat{R^{2n}} = \begin{pmatrix} R^n & 0 \\ f_n^m(R_1^n(n)) & R_1^n \\ f_n^m(R_2^n(n-1)) & R_1^n \\ f_n^m(R_3^n(n-2)) & R_1^n \\ \vdots & \vdots \\ f_n^m(R_n^n(1)) & R_1^n \\ 0 & R^n \end{pmatrix}, \qquad \widehat{U^{2n}} = \begin{pmatrix} U^n & 0 \\ & \\ 0 & U^n \end{pmatrix}$$

(m *es el número de columnas de* R^n*)*

(b) $f_k^m : \{0,1\}^m \longrightarrow [R^m \longrightarrow R^k]$, *es decir,* $f_k^m(v_1 \ldots v_m)$ *devuelve una matriz - una aplicación lineal - de dimensión* $k \times m$. f_k^m *se define de forma que las* k *filas son precisamente el vector argumento* $(v_1 \ldots v_m)$:

$$f_k^m(v_1 \ldots v_m) = \begin{pmatrix} v_1 & \ldots & v_m \\ \vdots & & \vdots \\ {\scriptstyle k)} & & \\ \vdots & & \vdots \\ v_1 & \ldots & v_m \end{pmatrix}$$

(c) *Refinamiento* ($\widehat{R^{2n}}, \widehat{U^{2n}}$) : *es un proceso consistente en dos fases:*

i. Pasos de ampliación: se ejecutan sobre la matriz $\widehat{R^{2n}}$ *tantos pasos de ampliación consecutivos como sean necesarios para conseguir que cada fila de los bloques* $f_n^m(R_i^n(n-i+1))$, $i = 1 \ldots n$, *así como de los bloques* R_1^n, tenga un sólo uno. *Los pasos de ampliación han de estar ligados a conjuntos de columnas* C *que incluyan* exclusivamente *columnas de los bloques izquierdos - los bloques de la forma* $f_n^m(R_i^n(n-i+1))$ *- o bien columnas de los bloques derechos - los de la forma* R_1^n.
Llamamos $\widehat{R'}^{2n}$, $\widehat{U'}^{2n}$ *a las matrices que resultan de la ejecución de estos* ***pasos de ampliación.***
En realidad la matriz $\widehat{\mathbf{U'}}^{\mathbf{2n}}$ ***es ya la matriz final*** $\mathbf{U^{2n}}$ *que queremos definir, pues no se ve afectada por la segunda fase del proceso de Refinamiento.*
Observación: probaremos en el Lema 17 que el número de ***pasos de ampliación*** *necesarios en la construcción de* R^{2n}, U^{2n} *es exactamente* $2\left[\frac{n}{2} + (\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1\right]$.

ii. Reordenación: probaremos en el Lema 13 que el producto de las matrices $\widehat{R^{2n}}$, $\widehat{U^{2n}}$ *es la matriz* S^{2n}, *de lo que se sigue que de la misma forma, el producto de las matrices* $\widehat{R'}^{2n}$, $\widehat{U'}^{2n}$ *es* S^{2n}, *pues hemos demostrado ya que los* ***pasos de ampliación*** *no afectan al producto de las matrices. Asumiendo este resultado, esta fase consiste en reordenar la matriz* $\widehat{R'}^{2n}$ *mediante la multiplicación* $I_{S^{2n} \to H^{2n}} \times \widehat{R'}^{2n}$, *donde* $I_{S^{2n} \to H^{2n}}$ *es la matriz que verifica* $I_{S^{2n} \to H^{2n}} \times S^{2n} = H^{2n}$. *Finalmente, la matriz* R^{2n} *que tratamos de definir, es precisamente* $\mathbf{R^{2n}} = \mathbf{I_{S^{2n} \to H^{2n}}} \times \widehat{\mathbf{R'}}^{\mathbf{2n}}$.
Observación: la existencia de la matriz $I_{S^{2n} \to H^{2n}}$ *se deduce trivialmente de la existencia de la matriz* $I_{H^{2n} \to S^{2n}}$, *lo que quedó establecido en el Corolario 4, pues si la expresión correspondiente a esta última es* $I^c_{i_1,j_1} \times I^c_{i_2,j_2} \times \ldots \times I^c_{i_k,j_k}$, *siendo* c *el número de filas de* H^{2n}, *entonces* $I_{S^{2n} \to H^{2n}} = I^c_{i_k,j_k} \times \ldots \times I^c_{i_2,j_2} \times I^c_{i_1,j_1}$.

El Lema 13 a continuación, es necesario para demostrar que nuestras matrices representan soluciones al *problema de los rangos variables*, resultado que

probaremos en el Teorema 6 inmediatamente después del lema.

Lema 13

Dadas las matrices $\widehat{R^{2n}}$, $\widehat{U^{2n}}$ *definidas como en el apartado* 2.*a de la definición 13, se verifica que*

$$\widehat{R^{2n}} \times \widehat{U^{2n}} = S^{2n}.$$

Demostración

Podemos hablar de *comprobación* de este resultado, mas que de *demostración*, pues se deduce trivialmente a partir de las definiciones de las matrices T^n, S^{2n}, H^{2n}, y de la propia definición de $\widehat{R^{2n}}$, $\widehat{U^{2n}}$, en la que - recuérdese - se asume recursivamente que el producto de los bloques componentes de la forma R^n, U^n es la matriz T^n ■

Teorema 6

Si R^n, U^n *son matrices definidas como en la Definición 13, entonces representan una solución al problema de los rangos variables de tamaño* n, *o lo que es lo mismo, verifican que* $R^n \times U^n = H^n$.

Demostración

El resultado se deduce trivialmente a partir de los Lemas 12 y 13, y de la definición de R^n, U^n ■

5.1.4 Complejidad y Número de Variables de Nuestras Matrices Solución

Los Teoremas 7 y 8 al final de este apartado establecen respectivamente la suma de los costes de las operaciones y el número de variables asociados a nuestras matrices solución.

Previamente necesitamos obtener una serie de resultados.

El Lema 14 a continuación, demuestra que ninguna fila de nuestras matrices R^n tiene más de dos unos. Esto significa que el coste máximo de una operación *Retrieve* es 2.

Lema 14

Sean las matrices R^n, U^n *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$, *con* n *de la forma* 2^k, $k \in \mathbf{N}^+$, *definidas como en la Definición 13.*

Entonces se verifica que el número máximo de unos presente en cada fila de R^n *es* 2, *sea cual sea el valor de* n.

Demostración

Demostramos el resultado por inducción. Lo suponemos cierto para $R^2 \ldots R^n$, y comprobamos que entonces es cierto para R^{2n}.

La matriz R^{2n} es el resultado de aplicar el proceso de refinamiento a la matriz $\widehat{R^{2n}}$, definida como

$$\widehat{R^{2n}} = \begin{pmatrix} R^n & 0 \\ f_n^m(R_1^n(n)) & R_1^n \\ f_n^m(R_2^n(n-1)) & R_1^n \\ f_n^m(R_3^n(n-2)) & R_1^n \\ \vdots & \vdots \\ f_n^m(R_n^n(1)) & R_1^n \\ 0 & R^n \end{pmatrix}$$

Por inducción sabemos que el número de unos por fila es como mucho 2 en los bloques de la forma $[\ R^n \mid 0\ ]$, $[\ 0 \mid R^n\ ]$. En cuanto al resto de bloques, el proceso de *Refinamiento* garantiza que en los bloques restantes el número de unos por fila es como mucho 2, pues se ejecutan los `pasos de ampliación` necesarios para garantizar que los bloques de la mitad izquierda de la matriz tienen un único uno por fila, al igual que los bloques de la mitad derecha de la matriz ■

El número de `pasos de ampliación` necesarios en la construcción de R^n, U^n, se calcula en el Lema 17. La demostración del Lema 17 requiere el establecimiento de dos lemas previos, el Lema 15 y el Lema 16 que se exponen a continuación. Ambos lemas son en cierta forma *simétricos* en el sentido de que establecen resultados análogos para las mitades izquierda y derecha de la matriz R^n.

Utilizamos la notación definida al comienzo de la sección 5.1.3, según la cual R_i^n denota al *i-ésimo* bloque de filas de R^n (dimensión $(n-i+1) \times m$ siendo

m el número de columnas de R^n) y $R_i^n(j)$ denota la *j-ésima* fila de dicho bloque.

Lema 15

Sean R^n, U^n *con* $n = 2^{k+1}$ *para un cierto natural* k, *matrices definidas como en la Definición 13.*

Se tiene que

1. *para todo* $i \in \left[(n - \frac{n}{2^r} + 2) \ldots (n - \frac{n}{2^{r+1}})\right]$, $r = 1 \ldots k-1$, *se verifica que* $\#R_i^n(n - i + 1) = 2$.

2. *para todo* $i = (n - \frac{n}{2^r} + 1)$, $r = 1 \ldots k$, *se verifica que* $\#R_i^n(n - i + 1) = 1$.

3. *para todo* $i = 1 \ldots \frac{n}{2}$ *se verifica que* $\#R_i^n(n - i + 1) = 2$.

Demostración

Demostramos el lema por inducción. El resultado es trivialmente cierto para $n = 2$, y asumiendo que es cierto para $n = 2^{k+1}$ lo demostramos para $2n$.

Tenemos que probar:

1. para todo $i \in \left[(2n - \frac{2n}{2^r} + 2) \ldots (2n - \frac{2n}{2^{r+1}})\right]$, $r = 1 \ldots k$, se verifica que $\#R_i^{2n}(2n - i + 1) = 2$. Distinguimos dos casos:

   (a) $r = 1$: hay que comprobar que para todo $i \in \left[(2n - n + 2) \ldots (2n - \frac{n}{2})\right]$ se tiene que $\#R_i^{2n}(2n - i + 1) = 2$.

   El número de filas de cada bloque R_i^{2n} es $(2n - i + 1)$, por tanto las dimensiones de los bloques R_i^{2n} con $i \in \left[(2n - n + 2) \ldots (2n - \frac{n}{2})\right] = [(n + 2) \ldots \frac{3n}{2}]$ oscilan entre $(\frac{n}{2} + 1)$ y $(n - 1)$. Como se observó en el Lema 10, las últimas $\frac{n(n+1)}{2}$ filas de R^{2n} son precisamente las últimas $\frac{n(n+1)}{2}$ filas de $\widehat{R^{2n}}$, y por tanto, por definición de $\widehat{R^{2n}}$, lo que tenemos que probar es que para todo $i = 2 \ldots \frac{n}{2}$ $\#R_i^n(n - i + 1) = 2$, pero eso es cierto por hipótesis de inducción (apartado 3. de la H.I).

   (b) $r = 2 \ldots k$: como en el caso anterior, nos estamos ocupando de filas de R^{2n} que son iguales a las filas correspondientes en $\widehat{R^{2n}}$, y por tanto lo que tenemos que probar en esta ocasión es que para

todo $i \in \left[(n - \frac{n}{2^r} + 2) \ldots (n - \frac{n}{2^{r+1}})\right]$, $r = 1 \ldots k-1$, se verifica que $\#R_i^n(n-i+1) = 2$, pero esto es cierto por el apartado 1. de la H.I.

2. para todo $i = (2n - \frac{2n}{2^r} + 1)$, $r = 1 \ldots k+1$, se verifica que $\#R_i^{2n}(2n - i + 1) = 1$. Distinguimos dos casos:

   (a) $r = 1$: hay que comprobar que para $i = (2n - n + 1) = n + 1$ se tiene que $\#R_{n+1}^{2n}(n) = 1$. Estamos tratando de una fila que es igual en $\widehat{R^{2n}}$ y en R^{2n}, pues forma parte de las $\frac{n(n+1)}{2}$ últimas filas. Pero $R_{n+1}^{2n}(n) = R_1^n(n)$, y dada la definición de $\widehat{R^{2n}}$, en la que los bloques centrales de la mitad derecha son precisamente de la forma R_1^n, tenemos la certeza de que si la fila $R_1^n(n)$ tuviese 2 unos, se le aplicaría un paso de ampliación. Esto garantiza que $\#R_{n+1}^{2n}(n) = 1$.

   (b) $r = 2 \ldots k+1$: de nuevo tenemos la certeza de que tratamos con filas situadas entre las $\frac{n(n+1)}{2}$ últimas, y por tanto son iguales en $\widehat{R^{2n}}$ y en R^{2n}. Por tanto debemos probar que para todo $i = (n - \frac{n}{2^r} + 1)$, $r = 1 \ldots k$, se verifica que $\#R_i^n(n-i+1) = 1$, , pero esto es precisamente el apartado 2. de la H.I.

3. para todo $i = 1 \ldots n$ se verifica que $\#R_i^{2n}(2n - i + 1) = 2$. Estamos hablando de las filas $R_1^{2n}(2n)$, $R_2^{2n}(2n-1)$, $R_3^{2n}(2n-2) \ldots R_n^{2n}(n+1)$, que corresponden a las operaciones $Retrieve(1, 2n)$, $Retrieve(2, 2n)$, $Retrieve(3, 2n) \ldots Retrieve(n, 2n)$. Tenemos que determinar que posición ocupaban estas filas en la matriz $\widehat{R^{2n}}$ (antes de aplicar el proceso de reordenación).Dada la correspondencia entre operaciones *Retrieve* y número de fila establecida en la definición 10, concluímos que dichas filas ocupaban en $\widehat{R^{2n}}$ posiciones correspondientes a los bloques *centrales*, es decir, bloques de la forma

$$[\, f_n^m(R_i^n(n-i+1)) \mid R_1^n \,]$$

y por tanto han de tener necesariamente dos unos.

La siguiente observación puede aclarar el propósito de este lema.

Observación 8

Obsérvese que

$$\left[\bigcup_{r=1}^{k-1}\bigcup_{i=(n-\frac{n}{2^r}+2)}^{(n-\frac{n}{2^{r+1}})} i\right]\bigcup\left[\bigcup_{r=1}^{k}(n-\frac{n}{2^r}+1)\right]\cup\{1\ldots\frac{n}{2}\}=[1\ldots(n-1)]$$

Esto quiere decir que el lema previo nos informa del número total de **pasos de ampliación** *que es necesario ejecutar en relación a la mitad izquierda de la matriz* $\widehat{R^{2n}}$ *en el proceso de Refinamiento que lleva finalmente a la obtención de* R^{2n}, U^{2n}, *pues vemos que estudia el total de bloques de la forma* $f_n^m(R_i(n-i+1))$ *que forman parte de la definición de* $\widehat{R^{2n}}$.

Lema 16

Sean R^n, U^n *con* $n=2^{k+1}$ *para un cierto natural* k, *matrices definidas como en la Definición 13.*

Se tiene que

1. *para todo* $i\in\left[(\frac{n}{2^{r+1}}+1)\ldots(\frac{n}{2^r}-1)\right]$, $r=1\ldots k-1$, *se verifica que* $\#R_1^n(i)=2$.
2. *para todo* $i=(\frac{n}{2^r})$, $r=1\ldots k$, *se verifica que* $\#R_1^n(i)=1$.
3. *para todo* $i=\frac{n}{2}+1\ldots n$ *se verifica que* $\#R_1^n(i)=2$.

Demostración

Omitimos la demostración, pues es similar - podríamos decir *simétrica* - a la utilizada para demostrar el Lema 15.

El propósito del Lema 16 se explica en la siguiente observación.

Observación 9

Obsérvese que

$$\left[\bigcup_{r=1}^{k-1}\bigcup_{i=(\frac{n}{2^{r+1}}+1)}^{(\frac{n}{2^r}-1)} i\right]\bigcup\left[\bigcup_{r=1}^{k}(\frac{n}{2^r})\right]\cup\{\frac{n}{2}+1\ldots n\}=[2\ldots n]$$

Esto significa que el Lema 16 establece el número total de **pasos de ampliación** *que es necesario ejecutar sobre la mitad derecha de la matriz* $\widehat{R^{2n}}$ *en el proceso*

de Refinamiento que lleva finalmente a la obtención de R^{2n}, U^{2n}, *pues vemos que estudia el total de filas - exceptuando la primera, que necesariamente tendrá un único 1 pues se corresponde con la operación* $Retrieve(1,1)$ *- del bloque* R_1^n *que forman parte de la definición de* $\widehat{R^{2n}}$.

Ahora podemos proceder a calcular el número de `pasos de ampliación` que requiere el proceso de *Refinamiento* de nuestras matrices solución.

Lema 17

Sean R^{2n}, U^{2n} *matrices definidas como en la Definición 13. Sea* $n = 2^{k+1}$ *para un cierto número natural* k.

El número de `pasos de ampliación` *que se ejecutan en la fase de Refinamiento del proceso de construcción de las matrices es* $2\left[\frac{n}{2} + (\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1\right]$.

Demostración

Probamos en primer lugar que el número de `pasos de ampliación` que afectan a los bloques izquierdos, de la forma $f_n^m(R_i^n(n-i+1))$, es exactamente $\frac{n}{2} + (\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1$.

Justificamos los términos de la expresión:

1. $\frac{n}{2}$: por el punto 3. del Lema 15 y por definición de $\widehat{R^{2n}}$.

2. $\frac{n}{2^{j+1}} - 1$, $j = 1 \ldots k-1$: por el punto 1. del Lema 15 y por definición de $\widehat{R^{2n}}$. Cada sumando de la forma $\frac{n}{2^{j+1}} - 1$ refleja el número de índices comprendidos en el intervalo $\left[(n - \frac{n}{2^j} + 2) \ldots (n - \frac{n}{2^{j+1}})\right]$.

Obsérvese que el punto 2. del Lema 15 garantiza que no es necesario realizar ningún otro `paso de ampliación` que afecte a la mitad izquierda de $\widehat{R^{2n}}$.

A continuación probamos que el número de `pasos de ampliación` que afectan a la mitad derecha de $\widehat{R^{2n}}$, formada por bloques de la forma R_1^n, es igualmente
$\frac{n}{2} + (\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1$.

Justificamos los términos de la expresión:

1. $\frac{n}{2}$: por el punto 3. del Lema 16 y por definición de $\widehat{R^{2n}}$.

2. $\frac{n}{2^{j+1}} - 1$, $j = 1 \ldots k-1$: por el punto 1. del Lema 16 y por definición de $\widehat{R^{2n}}$. Cada sumando de la forma $\frac{n}{2^{j+1}} - 1$ refleja el número de índices comprendidos en el intervalo $\left[(\frac{n}{2^{j+1}} + 1) \ldots (\frac{n}{2^j} - 1)\right]$.

el punto 2. del Lema 16 garantiza que no es necesario realizar ningún otro `paso de ampliación` que afecte a la mitad derecha de $\widehat{R^{2n}}$ ■

Podríamos preguntarnos cual es la finalidad de los pasos de ampliación que se ejecutan en la primera fase del proceso de *Refinamiento*. La respuesta es que, ejecutado bajo ciertas condiciones, un `paso de ampliacion` reduce el número de unos totales presentes en la pareja de matrices a las que se aplica. Comprobaremos que todos los `pasos de ampliacion` que se ejecutan en el proceso de *Refinamiento* reducen el número de unos.

El siguiente resultado establece la variación en el número de unos que implica un `paso de ampliación`.

Proposición 2

Sean A, B *dos matrices de ceros y unos tales que* $A \times B = S^n$, *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$.

La ejecución de un `paso de ampliacion` *asociado a los conjuntos* $C = \{c_1 \ldots c_l\}$, $F = \{f_1 \ldots f_q\}$ *de columnas y filas de* A *respectivamente, produce una variación del número total de unos presente en ambas matrices que responde a la expresión:*

$$\sum_{i \in C, j \in \{1 \ldots n\}} b_{i,j} + q - (l \times q).$$

Demostración

No tenemos más que fijarnos en la definición de `paso de ampliación` asociado a los conjuntos C, F dada en la Definición 12.

El sumando q es el numero de unos de la nueva columna z_0 que se inserta en A, la cantidad $l \times q$ es el número de unos que desaparecen de las columnas $\{c_1 \ldots c_l\}$ de A, y el sumando $\sum_{i \in C, j \in \{1 \ldots n\}} b_{i,j}$ representa la cantidad de unos de la nueva fila z_0 de B ■

Observación 10

Sean A, B *dos matrices de ceros y unos tales que* $A \times B = S^n$, *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$.

Si se ejecuta un `paso de ampliacion` *asociado a los conjuntos* $C = \{c_1 \dots c_l\}$, $F = \{f_1 \dots f_q\}$ *de columnas y filas de* A *respectivamente, tenemos que si* $\sum_{i \in C, j \in \{1 \dots n\}} b_{i,j} + q - (l \times q)$ *toma valor mayor que* 0 *entonces el número total de unos de las matrices aumenta; si el valor es igual a* 0 *entonces la cantidad de unos no varía, y si el valor es menor que* 0 *la cantidad de unos total disminuye.*

Corolario 5

Cada uno de los pasos de ampliación que se ejecutan en la fase de Refinamiento del proceso de construcción de nuestras matrices dado en la Definición 13, disminuye la suma total del número de unos presente en ambas matrices.

Demostración

Por la definición 13, tenemos que $< R^{2n}, U^{2n} > = Refinamiento(\widehat{R^{2n}}, \widehat{U^{2n}})$, siendo

$$\widehat{R^{2n}} = \begin{pmatrix} R^n & 0 \\ f_n^m(R_1^n(n)) & R_1^n \\ f_n^m(R_2^n(n-1)) & R_1^n \\ f_n^m(R_3^n(n-2)) & R_1^n \\ \vdots & \vdots \\ f_n^m(R_n^n(1)) & R_1^n \\ 0 & R^n \end{pmatrix}, \quad \widehat{U^{2n}} = \begin{pmatrix} U^n & 0 \\ & \\ 0 & U^n \end{pmatrix}$$

(m es el número de columnas de R^n)

Tal cual se describe en la Definición 13, los `pasos de ampliación` afectan de manera disjunta a los bloques de la forma $f_n^m(R_i^n(n-i+1))$ y a los bloques R_1^n.

Cada bloque $f_n^m(R_i^n(n-i+1))$ está formado por n filas iguales, con un máximo de 2 unos por fila (ver Lema 14), por lo que, de ser necesario un `paso de ampliación` afectará a un conjunto $C = \{c_1, c_2\}$ de columnas ($c_1, c_2 \leq m$) y a un conjunto de filas $F = \{f_1 \dots f_k\}$ donde $k \geq n+1$. Por definición de

`paso de ampliación` asociado a los conjuntos F y C (ver Definición 12), el número de unos en la primera de las matrices, en este caso, se reduce *al menos* en una cantidad $n+1$. Por otra parte, en la segunda de las matrices, $\widehat{U^{2n}}$, aparece una nueva fila suma de otras dos filas de índices menores o iguales que m, es decir, pertenecientes a la *mitad superior* de la forma $[\, U^n \mid 0 \,]$, por lo que la suma de sus unos es menor o igual que n (U^n tiene n columnas).

En cuanto al bloque R_1^n que se repite n veces en la *mitad derecha* de $\widehat{R^{2n}}$, el caso es similar al anterior, pues de ser necesario algún `paso de ampliación`, afectará de nuevo a un conjunto C de columnas con sólo dos elementos $\{c_1, c_2\}$ ($c_1, c_2 \geq m$) y a un conjunto de filas $F = \{f_1 \ldots f_k\}$ con $k \geq n+1$. El argumento en este caso es simétrico al utilizado en el caso anterior ■

A continuación, nos proponemos establecer el número de unos totales que suma una pareja de matrices como la que hemos definido. Para ello necesitamos probar el Lema 18 a continuación.

Lema 18

Sean $A = (a_{i,j})$, $B = (b_{i,j})$ *dos matrices de ceros y unos tales que* $A \times B = H^n$, *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$. *Sean* i, j *dos índices cualesquiera,* $1 \leq i \leq j \leq n$.

Se verifica que si $[\; a_{k,c} = 1 \leftrightarrow c \in \{c_1 \ldots c_s\} \;]$ *siendo* $k = \sum_{s=0}^{i-2}(n-s) + (j-i+1)$, *entonces* $\sum_{l \in \{c_1 \ldots c_s\}} \sum_{r=1}^{n} b_{l,r} = j-i+1$ *(* k *es la fila de* A *que* `corresponde` *a la operación Retrieve*(i,j)*).*

Demostración

Dado que el producto de las matrices es la matriz H^n y dada la correlación de valores entre los índices i, j y el número de fila k, se tiene que $a_{k,*} \times b_{*,c} = 1$ para las $(j-i+1)$ columnas c que verifican $i \leq c \leq j$ ($a_{k,*}$ denota la fila k de A y $b_{*,c}$ denota la columna c de B).

El resultado se deduce inmediatamente aplicando el Lema 11 que implica que las filas $\{c_1 \ldots c_s\}$ de la matriz B son necesariamente *disjuntas.* ■

Observación 11

El Lema 18 establece que la ejecución de un paso de ampliación afectando a una fila asociada *a la operación* $Retrieve(i,j)$ *para unos ciertos* i, j *(ver la correspondencia entre el número de fila y la operación Retrieve correspondiente en la Definición 10) supone un aumento del número de unos en la segunda de las matrices del par igual a* $j-i+1$ *(el efecto de un paso de ampliación en la segunda matriz, es la inserción de una nueva fila suma de otras).*

Teorema 7

Sean las matrices R^n, U^n *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$, *con* n *de la forma* 2^k, $k \in \mathbf{N}^+$, *definidas como en la Definición 13. Denotamos por* $\#R^n$, $\#U^n$ *al número de unos de las matrices* R^n *y* U^n *respectivamente.*

Se verifica que

$$\#R_n \ + \ \#U_n \ = \ \frac{3n^2}{2} - \frac{3n}{2}\log_2 n + \frac{9n}{2} - 2\log_2 n - 4 \ ,$$

lo que representa una complejidad media constante para el conjunto de las $n + \binom{n+1}{2}$ *operaciones Retrieve y Update.*

Demostración

Contamos por separado los unos de R^{2n} y de U^{2n} .

Suponemos que $n = 2^{k+1}$.

Llamamos $R(n) = \#R^n$ y tenemos:

$$R(2n) = 2R(n) - 2\left[\frac{n}{2} + (\frac{n}{2^2} - 1) + \ldots + (\frac{n}{2^k} - 1)\right] + 2n^2$$

Justificamos la ecuación recursiva antes de resolverla. Para deducir la ecuación tenemos que observar la definición de la matriz $\widehat{R^{2n}}$ dada en la Definición 13, y aplicar los Lemas 15 y 16. En principio se obtiene la expresión

$$R(2n) = 2R(n) + 2Termino1 - (n+1)Termino2$$

siendo

$$Termino1 = 2n\left[\frac{n}{2} + \sum_{r=1}^{k-1}\left(\frac{n}{2^{r+1}} - 1\right)\right] + nk$$

y

$$Termino2 = \frac{n}{2} + (\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1$$

El $Termino1$ cuenta - según lo establecido en el Lema 15 - el número de unos de los n bloques de la forma $f_n^m(R_i^n(n-i+1))$ que forman parte de la definición de $\widehat{R^{2n}}$. Dicho número de unos es igual - por lo establecido en el Lema 16 - al número de unos presentes en los n bloques de la forma R_1^n que también forman parte de la definición de $\widehat{R^{2n}}$, y esta es la causa del factor constante 2 que multiplica a dicho término.

El $Termino2$ expresa el número total de *pasos de ampliación* que se ejecutan sobre las matrices $\widehat{R^{2n}}$, $\widehat{U^{2n}}$. Dicho número se estableció en el Lema 17 . Cada *paso de ampliación* disminuye en $n+1$ el número de unos presente en la primera de las matrices, y de ahí el factor $(n+1)$ multiplicando a $Termino2$.

Se comprueba fácilmente que operando la expresión

$$2R(n) + 2Termino1 - (n+1)Termino2$$

se obtiene la expresión deseada

$$2R(n) - 2\left[\frac{n}{2} + (\frac{n}{2^2} - 1) + \ldots + (\frac{n}{2^k} - 1)\right] + 2n^2$$

y ya tenemos justificada la ecuación correspondiente a $R(2n)$.

La resolvemos (recordamos $n = 2^{k+1}$):

$$\begin{aligned}
R(2n) &= 2R(n) - 2\left[\frac{n}{2} + (\frac{n}{2^2} - 1) + \ldots + (\frac{n}{2^k} - 1)\right] + 2n^2 \\
&= 2R(n) - 2\left[-(k-1) + n\left(\frac{1}{2} + \frac{1}{2^2} + \ldots + \frac{1}{2^k}\right)\right] + 2n^2 \\
&= 2R(n) + 2(k-1) + 2n^2 - n\frac{\frac{1}{2^k} - 1}{\frac{1}{2} - 1} \\
&= 2R(n) + 2(k-1) + 2n^2 - 2n\left(1 - \frac{2}{n}\right) \\
&= 2R(n) + 2k - 2 + 2n^2 - 2n + 4 = 2\left[R(n) + k + n^2 - n + 1\right]
\end{aligned}$$

Introducimos un cambio de notación.Llamamos

$$\hat{R}(k+2) = R(2^{k+2})$$

y tenemos por tanto

$$\begin{aligned}\hat{R}(k+2) &= 2\left[\hat{R}(k+1)+k+2^{2k+2}-2^{k+1}+1\right] \\ &= 2\left[\hat{R}(k+1)+(k+1)+2^{k+1}(2^{k+1}-1)\right]\end{aligned}$$

Por tanto

$$\begin{aligned}\hat{R}(k+1) &= 2\left[\hat{R}(k)+k+2^k(2^k-1)\right] \\ &= 2\left[2\left[\hat{R}(k-1)+(k-1)+2^{k-1}(2^{k-1}-1)\right]+k+2^k(2^k-1)\right] \\ &= 2^2\left[\hat{R}(k-1)+(k-1)+2^{k-1}(2^{k-1}-1)\right]+2\left[k+2^k(2^k-1)\right] \\ &= 2^3\left[\hat{R}(k-2)+(k-2)+2^{k-2}(2^{k-2}-1)\right]+2^2\left[(k-1)+2^{k-1}(2^{k-1}-1)\right]+2\left[k+2^k\right. \\ &= 2\left[k+2^k(2^k-1)\right]+2^2\left[(k-1)+2^{k-1}(2^{k-1}-1)\right]+2^3\left[(k-2)+2^{k-2}(2^{k-2}-1)\right]+ \\ &\quad \ldots+2^k\left[\hat{R}(1)+1+2(2-1)\right] \\ &= \sum_{j=1}^{k} j2^{k-j+1}+2^{k+1}\sum_{j=1}^{k}(2^j-1)+4(2^k) \\ &= \sum_{j=1}^{k} j2^{k-j+1}+2^{k+1}\left[2^{k+1}-2-k\right]+4(2^k) \\ &= \sum_{j=1}^{k} j2^{k-j+1}+2^{2(k+1)}-k2^{k+1}\end{aligned}$$

Utilizamos el siguiente resultado, que probaremos inmediatamente después:

$$\sum_{j=1}^{k} j2^{k-j+1} = 2^{k+2}-2k-4$$

y obtenemos

$$\hat{R}(k+1) = 2^{2(k+1)}-k2^{k+1}+2^{k+2}-2k-4$$

y dado que $n=2^{k+1}$, tenemos

$$\begin{aligned}R(n) &= n^2-(\log_2 n-1)n+2n-2(\log_2 n-1)-4 \\ &= n^2-n\log_2 n+3n-2\log_2 n-2 \\ &= n^2+3n-(n+2)\log_2 n-2\end{aligned}$$

Nos queda por probar el resultado del que nos hemos ayudado, esto es:

$$\sum_{j=1}^{k} j2^{k-j+1} = 2^{k+2} - 2k - 4$$

Para ello, desarrollamos el término izquierdo de la igualdad:

$$\begin{aligned}
\sum_{j=1}^{k} j2^{k-j+1} &= 2^{k+1}\sum_{j=1}^{k} j2^{-j} \\
&= 2^{k+1}\left[\frac{1}{2} + 2\frac{1}{2^2} + 3\frac{1}{2^3} + \ldots + k\frac{1}{2^k}\right] \\
&= 2^{k+1}\left[\frac{\frac{1}{2^{k+1}} - \frac{1}{2}}{\frac{1}{2} - 1} + \left(\frac{1}{2^2} + 2\frac{1}{2^3} + \ldots + (k-1)\frac{1}{2^k}\right)\right] \\
&= 2^{k+1}\left[\left(1 - \frac{1}{2^k}\right) + \frac{\frac{1}{2^{k+2}} - \frac{1}{2^2}}{-\frac{1}{2}} + \left(\frac{1}{2^3} + 2\frac{1}{2^4} + \ldots + (k-2)\frac{1}{2^k}\right)\right] \\
&= 2^{k+1}\left[\left(1 - \frac{1}{2^k}\right) + \left(\frac{1}{2} - \frac{1}{2^k}\right) + \left(\frac{1}{2^2} - \frac{1}{2^k}\right) + \left(\frac{1}{2^4} + 2\frac{1}{2^5} + \ldots + (k-3)\frac{1}{2^k}\right)\right] \\
&= 2^{k+1}\left[\left(1 - \frac{1}{2^k}\right) + \left(\frac{1}{2} - \frac{1}{2^k}\right) + \left(\frac{1}{2^2} - \frac{1}{2^k}\right) + \ldots + \left(\frac{1}{2^{k-2}} - \frac{1}{2^k}\right) + \frac{1}{2^k}\right] \\
&= 2^{k+1}\left[\left(1 + \frac{1}{2} + \ldots + \frac{1}{2^{k-2}}\right) - (k-1)\frac{1}{2^k} + \frac{1}{2^k}\right] \\
&= 2^{k+1}\left[\frac{\frac{1}{2^{k-1}} - 1}{-\frac{1}{2}} - k\frac{1}{2^k} + \frac{1}{2^{k-1}}\right] \\
&= 2^{k+1}\left[2 - \frac{1}{2^{k-2}} - k\frac{1}{2^k} + \frac{1}{2^{k-1}}\right] \\
&= 2^{k+2} - 2^3 - 2k + 2^2 \\
&= 2^{k+2} - 2k - 4
\end{aligned}$$

como queríamos demostrar.

Calculamos ahora el número de unos de la matriz U^{2n}.

Llamamos $U(n) = \#U^n$, y tomando $n = 2^{k+1}$, tenemos:

$$U(2n) = 2U(n) + 2Termino1 + 2Termino2$$

donde ahora

$$Termino1 = n + (n-1) + \ldots + (\frac{n}{2} + 1)$$

y

$$Termino2 = \sum_{i=1}^{k-1} \left[(\frac{n}{2^i} - 1) + (\frac{n}{2^i} - 2) + \ldots + (\frac{n}{2^{i+1}} + 1) \right]$$

Justificamos esta ecuación recursiva.

$Termino1$ expresa el incremento de unos que produce cada *paso de ampliación* realizado en el proceso de construcción de las matrices R^{2n}, U^{2n} por adición de filas nuevas - que son suma de otras - en la segunda de estas matrices. Dicho incremento está calculado teniendo en cuenta el apartado 3. de los Lemas 15 y 16. Ambos apartados hacen referencia a filas de la primera matriz ligadas respectivamente a las operaciones $Retrieve(1,n), Retrieve(2,n) \ldots Retrieve(\frac{n}{2},n)$ y $Retrieve(1,\frac{n}{2}+1) \ldots Retrieve(1,n)$. Aplicando lo establecido en la Observación 11 en cuanto al incremento de unos en la segunda de las matrices que suponen los *pasos de ampliación* ligados a estas filas, se obtiene $Termino1$.

En cuanto al $Termino2$, representa lo mismo que $Termino1$ pero basándonos ahora en el apartado 1. de los Lemas 15 y 16. Dicho apartado hace referencia ahora a operaciones del tipo $Retrieve(i,n)$ y $Retrieve(1,i)$ respectivamente, con i variando sobre el conjunto de índices establecido en cada caso. De nuevo aplicamos lo establecido en la Observación 11 para calcular el aumento de unos que suponen los *pasos de ampliación* a realizar.

Por tanto tenemos que resolver la ecuación:

$$\begin{aligned} U(2n) &= 2U(n) + 2\left[n + (n-1) + \ldots + (\frac{n}{2}+1)\right] + \\ &\quad +2\left[\sum_{i=1}^{k-1}\left((\frac{n}{2^i} - 1) + (\frac{n}{2^i} - 2) + \ldots + (\frac{n}{2^{i+1}} + 1)\right)\right] \end{aligned}$$

Calculamos en primer lugar el valor de la expresión:

$$(*) = \left(n + (n-1) + \ldots + (\frac{n}{2}+1)\right) + \sum_{i=1}^{k-1}\left((\frac{n}{2^i} - 1) + (\frac{n}{2^i} - 2) + \ldots + (\frac{n}{2^{i+1}} + 1)\right)$$

y tenemos que es

$$\begin{aligned} (*) &= \frac{n(n+1)}{2} - 3 - \left[\frac{n}{2} + \frac{n}{2^2} + \ldots + \frac{n}{2^{k-1}}\right] \\ &= \frac{n(n+1)}{2} - 3 - \left[2^k + 2^{k-1} + \ldots + 2^2\right] \\ &= \frac{n(n+1)}{2} - 3 - (n-4) \end{aligned}$$

$$\begin{aligned} &= \frac{n(n+1)}{2} + 1 - n \\ &= \frac{n(n-1)}{2} + 1 \end{aligned}$$

y por tanto

$$U(2n) = 2U(n) + n(n-1) + 2$$

Introducimos un cambio de notación:

$$\hat{U}(k+1) = U(2^{k+1})$$

y la ecuación nos queda ahora

$$\begin{aligned} \hat{U}(k+1) &= 2\hat{U}(k) + 2^k(2^k - 1) + 2 \\ &= 2\left[2\hat{U}(k-1) + 2^{k-1}(2^{k-1} - 1) + 2\right] + 2^k(2^k - 1) + 2 \\ &= 2^2\left[2\hat{U}(k-2) + 2^{k-2}(2^{k-2} - 1) + 2\right] + 2^k\left(2^k + 2^{k-1}\right) - 2^{k+1} + 2^2 + 2 \\ &= 2^3\hat{U}(k-2) + 2^k\left(2^k + 2^{k-1} + 2^{k-2}\right) - 3(2^k) + \left[2^3 + 2^2 + 2\right] \\ &= \ldots \\ &= 2^k\hat{U}(1) + 2^k\left(2^k + 2^{k-1} + \ldots + 2\right) - k(2^k) + \left[2^k + \ldots + 2^2 + 2\right] \\ &= 2^{k+1} + 2^k\left(2^{k+1} - 2\right) - k2^k + \left(2^{k+1} - 2\right) \end{aligned}$$

Recuperamos la notación inicial obteniendo

$$\begin{aligned} U(n) &= n + \frac{n^2}{2} - n - (\log_2 n - 1)\frac{n}{2} + n - 2 \\ &= \frac{n^2}{2} - \frac{n}{2}\log_2 n + \frac{3n}{2} - 2 \end{aligned}$$

Finalmente, el resultado del Teorema se obtiene de la suma de las dos expresiones obtenidas:

$$\begin{aligned} R(n) + U(n) &= \left[n^2 + 3n - (n+2)\log_2 n - 2\right] + \left[\frac{n^2}{2} - \frac{n}{2}\log_2 n + \frac{3n}{2} - 2\right] \\ &= \frac{3n^2}{2} - \frac{3n}{2}\log_2 n + \frac{9n}{2} - 2\log_2 n - 4 \ \blacksquare \end{aligned}$$

A continuación, vamos a calcular el número de variables $z_1 \ldots z_m$ que requiere la solución definida por nuestras matrices, en función del tamaño n del problema. Llamamos $Var(n)$ a dicho número de variables, que como sabemos, coincide con el número de columnas y filas de R^n y U^n respectivamente.

Teorema 8

Sean las matrices R^n, U^n *de dimensiones respectivas* $\frac{n(n+1)}{2} \times m$ *y* $m \times n$, *con* n *de la forma* 2^k, $k \in \mathbf{N}^+$, *definidas como en la Definición 13.*

Se verifica que

$$m = n \log_2 n - 2n + 2 \log_2 n + 2$$

Demostración

Llamamos $V(n)$ al número de columnas de R^n, o lo que es lo mismo, al número de filas de U^n.

Consideramos $n = 2^k$.

Tenemos que

$$V(2n) \quad = \quad 2[V(n) + \frac{n}{2} + (\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1]$$

La ecuación responde a esa expresión por la definición de $\widehat{R^{2n}}$, $\widehat{U^{2n}}$ dada en la Definición 13 , y por el número de *pasos de ampliación* que se ejecuta sobre ellas, tal cual se estableció en el Lema 17. Como sabemos, cada *paso de ampliación* aumenta en 1 el número de filas/columnas .

Introducimos un cambio de notación, llamando $h(k) = V(2^k)$, y obtenemos

$$\begin{aligned}
h(k+1) &= 2[h(k) + 2^{k-1} + (2^{k-2} - 1) + \ldots + (2 - 1)] = 2[h(k) + (2^k - 2) - (k - 2)] \\
&= 2[h(k) + 2^k - k] = 2h(k) + 2^{k+1} - 2k \\
&= 2[2h(k-1) + 2^k - 2(k-1)] + 2^{k+1} - 2k \\
&= 2^2 h(k-1) + 2(2^{k+1}) - (2^k + 2^2(k-1)) \\
&= 2^3 h(k-2) + 3(2^{k+1}) - (2^k + 2^2(k-1) + 2^3(k-2)) \\
&= 2^k h(1) + k(2^{k+1}) - (2^k + 2^2(k-1) + 2^3(k-2) + \ldots + (2^k)1) \\
&= 2^k V(2) + k2^{k+1} - (2(k-1) + 2^2(k-2) + \ldots + (2^{k-1})1) - (2 + 2^2 + \ldots 2^k) \Rightarrow \\
V(2n) &= nV(2) + 2n \log_2 n - (2(k-2) + 2^2(k-3) + \ldots + (2^{k-2})1) \\
&\quad -(2 + 2^2 + \ldots + 2^{k-1}) - (2^{k+1} - 2) \\
&= nV(2) + 2n \log_2 n - (2(k-2) + 2^2(k-3) + \ldots + (2^{k-2})1) - (2^k - 2) - (2^{k+1} - 2) \\
&= nV(2) + 2n \log_2 n - (2^{k+1} - 2) - (2^k - 2) - (2^{k-1} - 2) - \ldots - [2.2 + 2^2.1] \\
&= nV(2) + 2n \log_2 n - (2^{k+1} - 2) - (2^k - 2) - (2^{k-1} - 2) - \ldots - (2^3 - 2) - (2^2 - 2) \\
&= nV(2) + 2n \log_2 n - (2^{k+2} - 2^2) + 2k
\end{aligned}$$

Finalmente tenemos

$$V(2n) = 2(n+1)\log_2 n - 2n + 4 \;\Rightarrow\; V(n) = n\log_2 n - 2n + 2\log_2 n + 2 \;\blacksquare$$

5.2 Modelo de Grafos para el Problema de los Rangos Variables

Para recordar el planteamiento original del *problema de los rangos variables*, se debe consultar el apartado 2.3.2 del Capítulo 2.

El objetivo de este apartado es establecer un modelo computacional basado en grafos para el estudio del *problema de los rangos variables*, y posteriormente definir grafos dirigidos que representan una solución para cualquier tamaño n del problema que se pueda expresar de la forma $n = 2^k,\ k \in \mathbf{N}$.

En el apartado 5.2.1 definimos el *modelo de grafos* para el estudio del problema.

También se define el concepto de `solución` asociada a un grafo. Para ello hay que definir los programas que implementan las operaciones *Update* y *Retrieve* y establecer el criterio que determina que un grafo represente una solución correcta para el problema de los rangos variables de acuerdo con tales programas. Finalmente se define la complejidad asociada a cada operación.

Posteriormente, en el apartado 5.2.2 definiremos nuestros grafos solución particulares. Comprobaremos que nuestro grafo para el problema de tamaño n, representa una solución *equivalente* a la definida por nuestras parejas de matrices R^n, U^n de la Definición 13 dentro del *modelo matricial* de la sección 5.1. El concepto de *equivalencia* supone que cada operación implica la consulta o modificación del mismo subconjunto de variables de programa, y por tanto tiene la misma complejidad. Es decir, nuestros grafos ofrecen la misma complejidad media para el total de las operaciones que la solución matricial propuesta en el apartado 5.1.3 de este mismo capítulo.

5.2.1 Definición del Modelo de Grafos

El modelo que proponemos para el estudio del *problema de los rangos variables*, incluye todos los programas que verifican:

- La estructura de datos solución agrupa en un vector a un conjunto de variables $Z = (z_1 \dots z_m)$ que toman valores en el semigrupo S.
- Unaa operación *Retrieve* se ejecuta sumando un subconjunto de variables de Z.
- Una operación *Update* se ejecuta incrementando un subconjunto de las variables de Z.

El modelo de grafos consiste en tuplas $< Z, G^n >$ donde $Z = \{z_1 \dots z_m\}$ es un conjunto de variables que toman valores en un semigrupo conmutativo S, y $G =< Nodos(G), Arcos(G) >$ es un grafo dirigido con $Nodos(G^n) = \{1 \dots m\}$ (el número de variables puede variar, aunque debe ser mayor o igual que n).

Asociados a una tupla $< Z, G^n >$, se definen los programas

Definición 14

Dada una tupla $< Z, G^n >$ *dentro del modelo de grafos para el problema de los rangos variables de tamaño* n, *definimos los siguientes algoritmos para la implementación de las operaciones Update y Retrieve:*

1. $Update(j, x)$: *para todo* $l \in Antecesores(j, G^n)$ *hacer* $[z_l \leftarrow z_l + x]$.
2. $Retrieve(i, j)$: *output* $\sum_{l \in Minimal(i,j,G^n)} z_l$, *siendo* $Minimal(i, j, G^n)$ *un conjunto de nodos de* G *de cardinal mínimo, tal que*

$$\bigcup_{w \in M_{i,j}} Sucesores(w, G^n) \cap \{1 \dots n\} = [i, i+1 \dots j]$$

$$\bigcap_{w \in M_{i,j}} [Sucesores(w, G^n) \cap \{1 \dots n\}] = \emptyset$$

donde

$M_{i,j} = Minimal(i, j, G^n)$

$Antecesores(j, G^n) = \{\ w\ /\ (w \in Nodos(G^n)) \wedge$ << *existe un camino que va de* w *a* j >> $\}$

$Sucesores(j, G^n) = \{\ w\ /\ (w \in Nodos(G^n)) \wedge$ << *existe un camino que va de* j *a* w >> $\}$

(el nodo j *se considera incluído tanto en el conjunto* $Antecesores(j, G^n)$ *como en* $Sucesores(j, G^n)$ *).*

Observación 12 *Mientras que las variables afectadas por la ejecución de una operación Update están unívocamente definidas, esto no ocurre en el caso de las operaciones* $Retrieve(i, j)$*. Efectivamente, podrían existir varios conjuntos de nodos del grafo tales que la unión de los Sucesores de los nodos del conjunto sea el intervalo cerrado* $[i, i+1 \ldots j]$*, y podría haber varios con el mínimo cardinal. En ese caso, se debería escoger uno cualquiera de ellos, aunque podemos fijar un criterio para que el procedimiento sea sistemático: por ejemplo, ordenando crecientemente los nodos del conjunto según la numeración del nodo, escogemos el conjunto minimal* menor *en cuanto al orden lexicográfico. En cualquier caso, la existencia del conjunto* $Minimal(i, j, G^n)$ *está garantizada teniendo en cuenta que*

- *En el Lema 19 se establece como exigencia para la corrección de los programas asociados a una tupla* $< Z, G^n >$*, el que para todo nodo* $w \in \{1 \ldots n\}$ *de* G^n *se tiene que* $Sucesores(w, G^n) \cap (\{1 \ldots n\}) = \{w\}$ *(recuérdese que consideramos que todo nodo es sucesor de si mismo).*

  Esto evita situaciones como la que refleja la figura 5.1 más abajo.

- *Como se dijo antes, el nodo* j *se considera incluído tanto en el conjunto* $Antecesores(j, G^n)$ *como en* $Sucesores(j, G^n)$*, y de no haber otro conjunto con menor cardinal, tendríamos que* $Minimal(i, j, G^n) = [i, i+1 \ldots j]$*. En este caso, el apartado anterior garantiza que la intersección de los sucesores de los nodos es el conjunto vacío.*

Traducido al modelo matricial de la sección 5.1, esta indeterminación se refleja en el hecho de que para una misma matriz B^n *pueden existir matrices distintas* $A_1^n \ldots A_k^n$ *tales que* $A_i^n \times B^n = H^n$, $A_i^n \neq A_j^n$ *si* $i \neq j$.

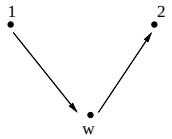


Figura 5.1: Parte de un grafo G^n con m nodos, siendo $2 < n < w \leq m$. El grafo no verifica la condición 1. que se exige para garantizar que representa una solución válida para el problema.

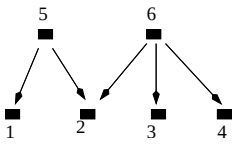


Figura 5.2: Grafo para el problema de tamaño 4. Retrieve(1,3) se puede ejecutar como *output* $z_3 + z_5$. Una opción distinta con coste mayor sería *output* $z_1 + z_2 + z_3$. De igual manera, la operación Retrieve(1,4) se puede ejecutar como *output* $z_1 + z_6$ y también como *output* $z_5 + z_3 + z_4$ o *output* $z_1 + z_2 + z_3 + z_4$, teniendo estas dos últimas opciones peor coste.

Ejemplo 3 *Veamos la pareja de matrices correspondiente al grafo de la figura 5.2, dentro del modelo matricial definido en la sección 5.1.*

La pareja de matrices A^4, B^4 *que se corresponden con la solución que*

expresa el grafo de la figura 5.2 es:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} = H^4$$

Vemos que con la misma matriz B^4, *es posible definir otra matriz* $(A^4)'$ *tal que el producto de ambas sigue siendo la matriz* H^4, *por lo que esta nueva pareja también representa una solución al problema de los rangos variables de tamaño* $n = 4$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} = H^4$$

La única diferencia entre A^4 *y* $(A^4)'$ *está en la tercera fila, que se corresponde con la operación* $Retrieve(1,3)$. *Ahora el efecto de esa operación es* [*output* $z_1 + z_2 + z_3$] *lo cual supone mayor complejidad de la operación, pues implica el acceso a tres variables de programa en lugar de a 2.*

Dada una tupla $< Z, G^n >$, el siguiente lema establece condiciónes sobre G^n que implican la corrección de los programas dados en la Definición 14.

Lema 19

Sea una tupla $< Z, G^n >$ *dentro del modelo de grafos para el problema de los rangos variables de tamaño* n. *Los programas dados en la Definición 14 son correctos si y sólo si*

1. *para todo* $w \in Nodos(G^n)$, $1 \leq w \leq n$, *se tiene que*

$$Sucesores(w, G^n) \cap (\{1 \ldots n\}) = \{w\}$$

2. *no existen ciclos en el grafo.*

Demostración

La demostración sigue el modelo de demostración utilizado en el Lema 1 de [14]. Necesitamos considerar el efecto de efectuar consecutivamente las operaciones

Retrieve(i,j), Update(r,x), Retrieve(i,j)

para cualesquiera $i, j, r \in \{1 \ldots n\}$, x elemento del semigrupo conmutativo cuyos valores se almacenan en las variables de Z.

Sean q_1, q_2 los output producidos como resultado de ejecutar las dos operaciones Retrieve(i,j), la primera y la segunda respectivamente.

A partir de la definición de las operaciones dada en el planteamiento general del problema de los rangos variables (ver apartado 2.3.2 del Capítulo 2), es obvio que los programas dados en la Definición 14 representan una solución correcta si y sólo si

$$q_2 - q_1 = \begin{cases} x & i \leq r \leq j \\ 0 & \text{e.o.c.} \end{cases} \qquad (Cond.1)$$

Sea $Minimal(i, j, G^n) = \{n1 \ldots n_k\}$, $k \geq 1$.

Por la Definición 14, tenemos que

$$q_1 = z_{n_1} + \ldots + z_{n_k}, \quad q_2 = q_1 + x(\ \#\ [\ Antecesores(r, G^n) \cap \{n1 \ldots n_k\}\ ]\ )$$

y por tanto la $Cond.(1)$ se verifica trivialmente a partir de la dos propiedades del conjunto $Minimal(i, j, G^n)$ enunciadas en 14 ■

Observación 13

Aunque no es una condición imprescindible para la corrección de los programas dados en la Definición 14, asumiremos que todo grafo G^n *de una tupla* $< Z, G^n >$ *solución del problema de los rangos variables de tamaño* n, *verifica que*

$$\forall\, w \in Nodos(G^n),\ \ [(w > n)\ \Rightarrow\ Sucesores(w, G^n) \cap \{1 \ldots n\} \neq \emptyset].$$

La existencia de nodos que no cumpliesen esa condición, tan sólo supondría la utilización de variables de programa adicionales que de hecho no intervendrían en las operaciones Update, Retrieve. Traducido al modelo matricial, si $A^n \times B^n = H^n$, *supondría la existencia de columnas de ceros en* A^n *y de filas de ceros en* B^n.

La exigencia de esta nueva condición, permite simplificar la segunda condición en la definición del conjunto $Minimal(i, j, G^n)$ *dada en la definición 14, que ahora podemos enunciar como :*

$$\Big(\bigcap Sucesores(w, G^n)\ /\ w \in Minimal(i, j, G^n) \Big) = \emptyset.$$

De hecho también implica una condición más fuerte que la condición primera del Lema 19, pues de hecho podemos concluir que en un grafo solución para el problema de los rangos variables de tamaño n, *no parten arcos de los nodos* $\{1 \ldots n\}$.

La siguiente observación muestra la correspondencia entre las soluciones al *problema de los rangos variables* dentro del *modelo de grafos* y dentro del marco formal establecido en [19] - se puede consultar en el apartado 2.3.2 del Capítulo 2.

Observación 14

Dada una tupla $< Z, G^n >$ *solución para el problema de los rangos variables de dimensión* n *dentro del modelo de grafos, con* $Z = (z_1 \ldots z_m)$, $Nodos(G^n) = \{1 \ldots m\}$, *y dados* $\{Y_1 \ldots Y_m\}$, $\{R_{i,j}\ /\ i, j = 1 \ldots n\}$ *definidos como en [19] (consultar apartado 2.3.2 del Capítulo 2), se tiene que*

- $\forall\, j = 1 \ldots n,\ \ [\, i \in Antecesores(j, G^n) \Leftrightarrow j \in Y_i\,]$

- $R_{i,j}$ *verifica:*

$$\bigcup_{w \in R_{i,j}} Sucesores(w, G^n) \cap \{1 \ldots n\} = [i, i+1 \ldots j]$$

$$\bigcap_{w \in R_{i,j}} [Sucesores(w, G^n) \cap \{1 \ldots n\}] = \emptyset$$

Obsérvese que, en particular, el conjunto $Minimal(i, j, G^n)$ *verifica estas propiedades. Esto simplemente incide en un hecho que hemos comentado anteriormente (vease la Observación 12) y es que un mismo grafo solución puede representar distintas alternativas en cuanto a la ejecución de una operación* $Retrieve(i, j)$.

A continuación definimos la complejidad asociada a las operaciones dentro del modelo de grafos.

Definición 15

Dada una tupla $< Z, G^n >$ *solución para el problema de los rangos variables de dimensión* n *dentro del modelo de grafos, con* $Z = (z_1 \ldots z_m)$, $Nodos(G^n) = \{1 \ldots m\}$, *definimos*

- *Complejidad asociada a la operación* $Retrieve(i, j)$: $\quad \# Minimal(i, j, G^n)$
- *Complejidad asociada a la operación* $Update(j, x)$: $\quad \# Antecesores(j, G^n)$

5.2.2 Definición de Grafos Solución Particulares. Complejidad Asociada y Número de Variables

Definimos un grafo cualquiera G, como una tupla $< Nodos(G), Arcos(G) >$, es decir, un conjunto de nodos y un conjunto de arcos,

En la definición de nuestros grafos solución utilizaremos un algoritmo que llamamos *Duplicacion*, que mostramos en la definición 16 a continuación.

Definición 16

Sea $< Z, G^n >$ *una tupla dentro del modelo de grafos para el problema de los rangos variables de tamaño* n, *tal que* $Nodos(G^n) = \{1 \ldots m\}$ *y tal que para todo* $i, j \in \{1 \ldots n\}$ *se verifica que* $\#Minimal(i, j, G) \leq 2$. *Definimos el procedimiento Duplicacion como:*

PROCEDURE Duplicacion (G^n : *in* `Grafo`, G^{2n} : *out* `Grafo`*)*
VAR *nuevosNodos:* `conjuntoNodos`, *nuevosArcos:* `conjuntoArcos`
BEGIN
$nuevosNodos := \{1 \ldots 2m\}$;
$nuevosArcos := \emptyset$;
forall $< v_1 \rightarrow v_2 > \in Arcos(G^n)$ *loop* — *Bucle 1*
if $v2 \leq n$ *then*
$nuevosArcos := nuevosArcos \cup \{ < (v_1 + n) \rightarrow v_2 > \} \cup$
$\{ < (v_1 + m) \rightarrow (v_2 + n) > \}$
else
$nuevosArcos := nuevosArcos \cup \{ < (v_1 + n) \rightarrow (v_2 + n) > \} \cup$
$\{ < (v_1 + m) \rightarrow (v_2 + m) > \}$
end if;
end loop;
$otroNodo := 2m$;
for $i = 1$ *to* $(n - 1)$ *loop* — *Bucle 2*
if $\#Minimal(i, n) = 2$ *then*
$otroNodo := otroNodo+1$;
$nuevosNodos := nuevosNodos \cup otroNodo$;
let $\{n_1, n_2\} \leftarrow Minimal(i, n)$ *in*
$nuevosArcos := nuevosArcos \cup \{ < otroNodo \rightarrow n_1 > \} \cup$
$\{ < otroNodo \rightarrow n_2 > \}$
<< *obsérvese que tras esta instrucción, se tiene que* $Minimal(i, n) = otroNodo$ >>
end if;
end loop;

for $i = 2n$ *downto* $(n+2)$ *loop* — *Bucle 3*

if $\#Minimal(n+1,i) = 2$ *then*

otroNodo := *otroNodo+1;*

nuevosNodos := *nuevosNodos* $\cup$ *otroNodo;*

let $\{n_1, n_2\} \leftarrow Minimal(n+1,i)$ *in*

nuevosArcos := *nuevosArcos* $\cup$ $\{ < otroNodo \rightarrow n_1 > \} \cup$

$\{ < otroNodo \rightarrow n_2 > \}$

<< *tras esta instrucción se tiene que* $Minimal(n+1,i) = otroNodo$ >>

end if;

end loop;

$G^{2n} =< nuevosNodos, nuevosArcos >;$

END.

Las siguientes figuras muestran grafos de distintos tamaños obtenidos mediante el procedimiento de Duplicación.

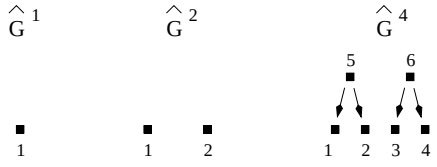


Figura 5.3: Grafos $\hat{G}^1$, $\hat{G}^2$ y $\hat{G}^4$.

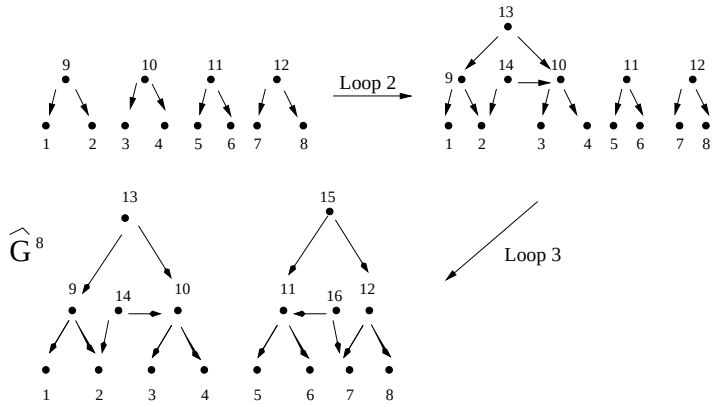


Figura 5.4: La figura muestra los grafos resultantes de ejecutar el *Bucle 1* , *Bucle 2* y *Bucle 3* del proceso de **Duplicación** respectivamente. Tras la ejecución del *Bucle 3* hemos obtenido finalmente el grafo $\hat{G}^8$.

A continuación, demostramos que si el procedimiento `Duplicacion` se aplica a un grafo G que representa una solución correcta al *problema de los rangos variables* de tamaño n, y que además cumple la propiedad adicional mencionada en la Observación 13, y la propiedad exigida en la propia definición del proceso de `Duplicacion`, esto es, que para cualesquiera índices $i, j \in \{1 \ldots n\}$ se tiene $Minimal(i, j, G) \leq 2$, entonces el grafo resultado representa una solución al *problema de los rangos variables* de tamaño $2n$ y verifica igualmente esas dos condiciones.

Lema 20 *Sea G^n un grafo que representa una solución correcta para el problema de los rangos variables de tamaño n. Suponemos que verifica la observación 13, que su conjunto de nodos es $Nodos(G^n) = \{1 \ldots m\}$ y que para todo $i, j \in \{1 \ldots n\}$ se verifica que $\#Minimal(i, j, G) \leq 2$.*

Sea G^{2n} el grafo resultado de aplicar el algoritmo de `Duplicacion` al grafo G^n.

Entonces G^{2n} representa una solución correcta para el problema de los rangos variables de tamaño $2n$, verificando además la propiedad enunciada en la Observación 13 y también que para todo $i, j \in \{1 \ldots 2n\}$, $\#Minimal(i, j, G^{2n}) \leq 2$.

Demostración

Como sabemos por el Lema 19, un grafo G cuyo conjunto de nodos es $\{1 \ldots m\}$, con $n \leq m$, es solución del *problema de los rangos variables* de tamaño n de acuerdo con la implementación de las operaciones *Update* y *Retrieve* dada en la definición 14, si y sólo si no tiene ciclos y los nodos $\{1 \ldots n\}$ no tienen sucesor (distinto de si mismo).

Tenemos que comprobar que G^{2n} no tiene ciclos y que los nodos $\{1 \ldots 2n\}$ no tienen sucesor distinto de si mismo. Analizamos el código del procedimiento `Duplicacion` que aparece en la definición 16.

El *Bucle 1* genera un grafo formado por dos subgrafos que son iguales al grafo G^n, excepto en la numeración de los nodos.

En uno de ellos, los nodos sin sucesor continúan siendo los numerados con $\{1 \ldots n\}$, y la numeración de los nodos con sucesor está *incrementada* en n respecto a la numeración que tenían en G^n. Es decir, si en G^n un nodo con sucesor estaba numerado como i, en este subgrafo aparecerá numerado como $i + n$. Este subgrafo se genera con los fragmentos de código de la sentencia *if-then-else* que añaden los arcos $< (v_1 + n) \rightarrow v_2 >$ y $< (v_1 + n) \rightarrow (v_2 + n) >$ a partir de cada arco $< v_1 \rightarrow v_2 >$ de G^n.

En el otro subgrafo, los n nodos que no tienen sucesor (distinto de si mismo) son efectivamente los numerados como $\{(n+1) \ldots 2n\}$. Los nodos con sucesor estan *desplazados* según un incremento de m respecto a su numeración en G^n. Es decir, si en G^n un nodo con sucesor estaba numerado como i, en este segundo subgrafo aparecerá numerado como $i + m$. Este subgrafo se genera con los fragmentos de código de la sentencia *if-then-else* que añaden los arcos $< (v_1 + m) \rightarrow (v_2 + n) >$ y $< (v_1 + m) \rightarrow (v_2 + m) >$ a partir de cada arco $< v_1 \rightarrow v_2 >$ de G^n.

En conclusión, el *Bucle 1* no afecta en cuanto a las dos propiedades que determinan la corrección de la solución, y asumiendo que se cumplen para G^n se concluye que siguen verificándose.

En cuanto al los bucles *2* y *3*, añaden arcos que tienen como origen nuevos nodos de numeración estrictamente mayor que $2m$, y como destino nodos ya existentes, y por tanto tampoco afectan a la existencia o no de ciclos, al hecho de que los nodos $\{1 \ldots 2n\}$ tengan sucesor o al cumplimiento de la propiedad

establecida en la observación 13.

El que $\#Minimal(i,j,G^{2n}) \leq 2 \quad \forall i,j\ 1 \leq i \leq j \leq n$, se deduce trivialmente a partir del código dado en 16 ■

Probamos ahora una proposición que justifica los bucles 2 y 3 de la definición 16. La adición de nodos según los criterios que rigen ambos bucles, pretende mejorar la suma de los costes de las $n + \frac{n(n+1)}{2}$ posibles operaciones *Update* y *Retrieve*.

Proposición 3

La adición de un nuevo nodo en los bucles 2 y 3 del procedimiento Duplicacion de la Definición 16, disminuye la suma total de las complejidades de las operaciones Update y Retrieve.

Demostración

Nos fijamos en el código que describe el proceso de `Duplicacion` de un grafo G^n solución para el *problema de los rangos variables* de tamaño n.

La ejecución de la instrucción interna del *Bucle 2* para un cierto índice i_0, incrementa (en caso de que la condición del *if-then* sea cierta) en una unidad la complejidad de las operaciones $Update(i_0, \_), Update(i_0{+}1, \_) \ldots Update(n, \_)$, es decir, supone un aumento de $(n - i_0 + 1)$ en la suma total de complejidades de las operaciones. Pero al mismo tiempo, garantiza un ahorro de 1 en la ejecución de todas aquellas operaciones $Retrieve(i_0, j)$ tales que $n \leq j \leq 2n$, lo que supone un ahorro seguro de $(n + 1)$, y por tanto la adición del nuevo nodo *compensa* en cualquier caso.

El razonamiento es simétrico para el *Bucle 3.* ■

Procedemos ahora a definir nuestros grafos solución.

Llamaremos $\hat{G}^n$ a nuestro grafo solución para el *problema de los rangos variables* de tamaño n.

Damos una definición recursiva de los mismos:

Definición 17 *Sea* $n = 2^k,\ k \in \mathbf{N}^+$. *Definimos el grafo* $\hat{G}^n$ *como:*

1. *si* $n = 2$: $\hat{G}^2 = < \{1,2\}, \emptyset >$.

2. si $n \Rightarrow 2n$: $\hat{G}^{2n} = Duplicacion(\hat{G}^n)$.

Teorema 9

Para todo $n = 2^k \quad k \in \mathbf{N}^+$, *el grafo* $\hat{G}^n$ *de la Definición 17 representa una solución correcta al problema de los rangos variables de tamaño* n.

Demostración

El resultado es inmediato a partir del caso base de la recursión y del Lema 20 ■

Corolario 6

Para todo $n = 2^k \quad k \in \mathbf{N}^+$, *el grafo* $\hat{G}^n$ *de la Definición 17 representa una solución correcta al problema de los rangos variables de tamaño* n *tal que la complejidad de cualquier operación* $Retrieve(i,j) \quad 1 \leq i \leq j \leq n$, *es menor o igual que 2.*

Demostración

El resultado es inmediato a partir de la definición 17 y del Lema 20, en el que se establece que si un grafo se genera mediante el proceso de `Duplicacion` a partir de otro que cumple que la complejidad de toda operación *Retrieve* - es decir, el cardinal del conjunto *Minimal* asociado a la operación - es 2, entonces el grafo generado también cumple dicha propiedad ■

A continuación vamos a calcular el número de nodos de $\hat{G}^n$, es decir el número de variables $z_1 \ldots z_m$ que requiere la solución definida por nuestros grafos, en función del tamaño n del problema. Llamamos $V(n)$ al número de nodos del grafo $\hat{G}^n$.

Necesitamos establecer un lema previo.

Lema 21

Sea $\hat{G}^n$, *el grafo de la Definición 17, con* $n = 2^{k+1}$, $k \in \mathbf{N}$. *Se verifica:*

1. $\neg\exists\ w \in Nodos(\hat{G}^n)\ /\ Sucesores(w, \hat{G}^n) \cap [1 \ldots n] = [i \ldots n]$ *con* $i \in \left[(n - \frac{n}{2^r} + 2) \ldots (n - \frac{n}{2^{r+1}})\right], \quad r = 1 \ldots k-1.$

2. $\exists\ w \in\ Nodos(\hat{G}^n)\ /\ Sucesores(w, \hat{G}^n) \cap [1 \ldots n] = [i \ldots n] \quad \forall\ i = (n - \frac{n}{2^r} + 1), \quad r = 1 \ldots k.$

Demostración

Procedemos por inducción. La comprobación es trivial para $n = 2$. Asumimos que el resultado es cierto para $n = 2^{k+1}$ y lo probamos para $2n$. Tenemos que comprobar:

1. $\neg\exists\ w \in\ Nodos(\hat{G}^{2n})\ /\ Sucesores(w, \hat{G}^{2n}) \cap [1 \ldots 2n] = [i \ldots 2n]$ con $i \in \left[(2n - \frac{2n}{2^r} + 2) \ldots (2n - \frac{2n}{2^{r+1}})\right], \quad r = 1 \ldots k.$

   El grafo $\hat{G}^{2n}$ se forma inicialmente a partir de dos subgrafos independientes iguales a $\hat{G}^n$ salvo en la numeración de los nodos. Nos ocupamos en este caso del subgrafo cuya numeración de los nodos sin sucesor es $[n + 1 \ldots 2n]$. La adición de nuevos nodos conectados a este subgrafo mediante los correspondientes arcos, se realiza en el *Bucle 3* del procedimiento de `Duplicación`. En dicho bucle se añaden exclusivamente nodos *cubriendo* un intervalo de nodos $[n+1 \ldots j]$ para algún $n+2 \leq j \leq 2n$, lo cual no afecta al cumplimiento de la propiedad que estamos analizando.

   Distinguimos los casos:

   - $r = 1$: se trata de ver que $\neg\exists\ w \in\ Nodos(\hat{G}^{2n})\ /\ Sucesores(w, \hat{G}^{2n}) \cap [1 \ldots 2n] = [i \ldots 2n]$ con $i \in \left[(2n - n + 2) \ldots (2n - \frac{n}{2})\right]$. Basta con comprobar que en $\hat{G}^n$ no existe un w que *cubra* el intervalo $[i \ldots n]$ con $i \in [2 \ldots (n - \frac{n}{2})]$. Pero es trivialmente cierto que para cualquier n, el grafo $\hat{G}^n$ no contiene nodos cuyo conjunto de sucesores incluya nodos del intervalo $[1 \ldots \frac{n}{2}]$ y del intervalo $[\frac{n}{2} + 1 \ldots n]$.
   - $r = 2 \ldots k$: de nuevo dada la observación realizada inmediatamente antes de la distinción en casos, nos basta con comprobar que en $\hat{G}^n$ no existe un w que *cubra* el intervalo $[i \ldots n]$ con $i \in [(n - \frac{n}{2^r} + 2) \ldots (n - \frac{n}{2^{r+1}})]$, siendo $r = 1 \ldots k - 1$, pero esto es precisamente la hipótesis de inducción.

2. $\exists\ w \in\ Nodos(\hat{G}^{2n})\ /\ Sucesores(w, \hat{G}^{2n}) \cap [1 \ldots 2n] = [i \ldots 2n] \quad \forall\ i = (2n - \frac{2n}{2^r} + 1), \quad r = 1 \ldots k + 1.$

   De nuevo vamos a distinguir dos casos:

- $r = 1$: se trata de ver que existe un nodo w en $\hat{G}^{2n}$ que *cubra* el intervalo de nodos sin sucesor $[(2n - n + 1) \ldots 2n]$, lo cual es trivialmente cierto pues se añade en el *Bucle 3* del proceso de `Duplicación` cuando el valor del índice del bucle es $2n$.
- $r = 2 \ldots k+1$: hay que comprobar que ya en $\hat{G}^n$ existe un nodo w que *cubre* el intervalo de nodos sin sucesor $[(n - \frac{n}{2^r} + 1) \ldots n] \quad \forall \quad r = 1 \ldots k$, o en caso contrario, que dicho nodo se añade en el proceso de `Duplicación` de $\hat{G}^n$ para obtener $\hat{G}^{2n}$. Esto último no es necesario, pues la primera condición es directamente la hipótesis de inducción ■

Enunciamos un lema *simétrico* a este que utilizaremos igualmente en el cálculo del número de nodos de nuestros grafos.

Lema 22

Sea $\hat{G}^n$, *el grafo de la Definición 17, con* $n = 2^{k+1}$, $k \in \mathbf{N}$. *Se verifica:*

1. $\neg\exists\ w \in Nodos(\hat{G}^n)\ /\ Sucesores(w, \hat{G}^n) \cap [1 \ldots n] = [1 \ldots i]$ *con* $i \in \left[(\frac{n}{2^{r+1}} + 1) \ldots (\frac{n}{2^r} - 1)\right], \quad r = 1 \ldots k - 1.$
2. $\exists\ w \in Nodos(\hat{G}^n)\ /\ Sucesores(w, \hat{G}^n) \cap [1 \ldots n] = [1 \ldots i] \quad \forall\ i = (\frac{n}{2^r}), \quad r = 1 \ldots k.$

Demostración

La demostración se puede realizar por inducción, de forma similar a la empleada en el Lema 21, y no consideramos necesaria su inclusión ■

Calculamos ahora el número de nodos del grafo $\hat{G}^n$.

Teorema 10

Sea el grafo $\hat{G}^n$, *con* $n = 2^k$ *para un cierto* $k \in \mathbf{N}^+$, *definido como en la Definición 17.*

Se verifica que

$$\#Nodos(\hat{G}^n) \ = \ n \log_2 n - 2n + 2 \log_2 n + 2$$

Demostración

Llamamos $V(n) = \#Nodos(\hat{G}^n)$.

Tenemos que

$$V(2n) \quad = \quad 2[V(n) + \frac{n}{2} + (\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1]$$

Esta desigualdad se deduce de la definición de nuestros grafos, del algoritmo de `Duplicacion` y de los Lemas 21 y 22.

El sumando $2V(n)$ proviene del hecho de que el grafo $\hat{G}^{2n}$ se genera a partir de dos subgrafos independientes iguales a $\hat{G}^n$ salvo en la numeración de los nodos.

El factor 2 que multiplica al resto de sumandos del corchete refleja la simetría en el proceso de adición de nodos y arcos en cada uno de estos subgrafos.

Explicamos la aparición de los términos del corchete refiriéndonos al subgrafo cuyo conjunto de nodos sin sucesor es $\{1 \ldots n\}$:

- el sumando $\frac{n}{2}$ se justifica por el hecho de que en el grafo $\hat{G}^n$ no existen nodos cuyo conjunto de sucesores incluya nodos del intervalo $[1 \ldots \frac{n}{2}]$ y del intervalo $[\frac{n}{2} + 1 \ldots n]$ simultáneamente. El proceso de `Duplicación` añade tales nodos en el *Bucle 2*, que garantiza que tras su ejecución se tiene que para todo $i = 1 \ldots (n-1)$, y por tanto para los $i \in \{1 \ldots \frac{n}{2}\}$, existe un nodo w que *cubre* el intervalo de nodos sin sucesor $[i \ldots n]$.

- la suma de términos $(\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1$ se justifica por el apartado (1) del Lema 21, en el que se afirma que en $\hat{G}^n$ no existen nodos que cubran los intervalos de nodos sin sucesor de la forma $[i \ldots n]$ para los $i \in \left[(n - \frac{n}{2^r} + 2) \ldots (n - \frac{n}{2^{r+1}})\right]$, $r = 1 \ldots k-1$ (considerando $n = 2^{k+1}$), por lo que tales nodos han de ser añadidos en el *Bucle 2* del proceso de `Duplicación`. Así, el sumando de la forma $(\frac{n}{2^{j+1}} - 1)$ refleja el número de nodos que incluye el intervalo $\left[(n - \frac{n}{2^j} + 2) \ldots (n - \frac{n}{2^{j+1}})\right]$.

La explicación a referida al subrafo derecho de $\hat{G}^{2n}$ es la siguiente:

- el sumando $\frac{n}{2}$ se justifica de nuevo por el hecho de que en el grafo $\hat{G}^n$ no existen nodos cuyo conjunto de sucesores incluya nodos del intervalo $[1 \ldots \frac{n}{2}]$ y del intervalo $[\frac{n}{2} + 1 \ldots n]$ simultáneamente. Esto implica

que en $\hat{G}^{2n}$ los nodos que cubren el intervalo $[n+1 \ldots i]$ para todo $i = (2n - \frac{n}{2} + 1) \ldots 2n$, deben ser añadidos en el *Bucle 3* del proceso de `Duplicación`.

- la suma de términos $(\frac{n}{2^2}-1)+(\frac{n}{2^3}-1)+\ldots+1$ se justifica de forma similar a la empleada en el subgrafo izquierdo, pero aplicando ahora el Lema 22, simétrico del Lema 21. Los nodos no existentes en $\hat{G}^n$ señalados en el apartado (1) de dicho lema, han de ser añadidos en el *Bucle 3* del proceso de `Duplicación` que genera $\hat{G}^{2n}$ a partir de $\hat{G}^n$.

Una vez justificada la ecuación, procedemos a resolverla considerando ahora que $n = 2^k$, para un cierto número natural $k \geq 1$ y teniendo en cuenta que $V(2) = 2$. Tenemos

$$V(2n) \quad = \quad 2[V(n) + \frac{n}{2} + (\frac{n}{2^2} - 1) + (\frac{n}{2^3} - 1) + \ldots + 1]$$

Introducimos un cambio de notación, llamando $h(k) = V(2^k)$, y obtenemos

$$\begin{aligned}
h(k+1) &= 2[h(k) + 2^{k-1} + (2^{k-2} - 1) + \ldots + (2-1)] = 2[h(k) + (2^k - 2) - (k-2)] \\
&= 2[h(k) + 2^k - k] = 2h(k) + 2^{k+1} - 2k \\
&= 2[2h(k-1) + 2^k - 2(k-1)] + 2^{k+1} - 2k \\
&= 2^2 h(k-1) + 2(2^{k+1}) - (2^k + 2^2(k-1)) \\
&= 2^3 h(k-2) + 3(2^{k+1}) - (2^k + 2^2(k-1) + 2^3(k-2)) \\
&= 2^k h(1) + k(2^{k+1}) - (2^k + 2^2(k-1) + 2^3(k-2) + \ldots + (2^k)1) \\
&= 2^k V(2) + k2^{k+1} - (2(k-1) + 2^2(k-2) + \ldots + (2^{k-1})1) - (2 + 2^2 + \ldots 2^k) \Rightarrow \\
V(2n) &= nV(2) + 2n\log_2 n - (2(k-2) + 2^2(k-3) + \ldots + (2^{k-2})1) \\
&\quad -(2 + 2^2 + \ldots + 2^{k-1}) - (2^{k+1} - 2) \\
&= nV(2) + 2n\log_2 n - (2(k-2) + 2^2(k-3) + \ldots + (2^{k-2})1) - (2^k - 2) - (2^{k+1} - 2) \\
&= nV(2) + 2n\log_2 n - (2^{k+1} - 2) - (2^k - 2) - (2^{k-1} - 2) - \ldots - [2.2 + 2^2.1] \\
&= nV(2) + 2n\log_2 n - (2^{k+1} - 2) - (2^k - 2) - (2^{k-1} - 2) - \ldots - (2^3 - 2) - (2^2 - 2) \\
&= nV(2) + 2n\log_2 n - (2^{k+2} - 2^2) + 2k
\end{aligned}$$

Finalmente tenemos

$$V(2n) = 2(n+1)\log_2 n - 2n + 4 \quad \Rightarrow \quad V(n) = n\log_2 n - 2n + 2\log_2 n + 2 \quad \blacksquare$$

A continuación calculamos la suma de los costes de las $n + \frac{n(n+1)}{2}$ operaciones *Update*, *Retrieve* posibles sobre un grafo $\hat{G}^n$ definido como en 17.

Teorema 11

Sea el grafo $\hat{G}^n$, *con* $n = 2^k$ *para un cierto* $k \in \mathbf{N}^+$, *definido como en 17. Denotamos por*

$$\#Retrieve^n = \sum_{1 \leq i \leq j \leq n} complejidad(Retrieve(i,j))$$

$$\#Update^n = \sum_{1 \leq j \leq n} complejidad(Update(j, \_))$$

Entonces $\#Retrieve^n + \#Update^n = \frac{3n^2}{2} - \frac{3n}{2}\log_2 n + \frac{9n}{2} - 2\log_2 n - 4.$

Demostración

Analizamos por separado la suma de costes de las operaciones *Retrieve* y las operaciones *Update*.

Llamamos $R(n) = \#Retrieve^n$. Consideramos que $n = 2^{k+1}$ y tenemos:

$$R(2n) = 2R(n) - 2\left[\frac{n}{2} + (\frac{n}{2^2} - 1) + \ldots + (\frac{n}{2^k} - 1)\right] + 2n^2$$

Justificamos la ecuación recursiva antes de resolverla.

El grafo $\hat{G}^{2n}$ se forma inicialmente a partir de dos subgrafos iguales a $\hat{G}^n$ salvo en la numeración de los nodos. De ahí el sumando $2R(n)$.

Posteriormente se lleva a cabo un proceso de adición de nodos y arcos cuya finalidad es abaratar el coste de ciertas operaciones *Retrieve* (bucles 2 y 3 del proceso de `Duplicación`), y de ahí el término que aparece restando. La constante 2 multiplicando dicho término expresa la simetría en el proceso de adición de arcos y nodos sobre los dos subgrafos iniciales. La adición de cada nuevo nodo, abarata exactamente en 1 el coste de una operación *Retrieve* concreta, por tanto debemos restar exactamente la cantidad de nodos añadidos en el proceso de `Duplicación` de $\hat{G}^n$ para obtener $\hat{G}^{2n}$. Tal como se demuestra en el Teorema 10, el término que aparece restando en el lado derecho de la ecuación es precisamente dicha cantidad.

En cuanto al sumando $2n^2$, expresa el número de operaciones $Retrieve(i,j)$ tales que $1 \leq i \leq n < j \leq 2n$. Dicho número es exactamente n^2, y cada una de dichas operaciones tiene coste 2 (ver Corolario 6).

Resolviendo la ecuación:

$$R(2n) \quad = \quad 2R(n) - 2\left[\frac{n}{2} + (\frac{n}{2^2} - 1) + \ldots + (\frac{n}{2^k} - 1)\right] + 2n^2$$

se obtiene

$$R(n) \quad = \quad \left[n^2 + 3n - (n+2)\log_2 n - 2\right]$$

tal cual se demostró en el Teorema 7 en el apartado correspondiente al modelo matricial para el problema de los rangos variables.

Calculamos ahora la suma de costes de las operaciones *Update*. Llamamos $U(n) = \#Update^n$. Consideramos que $n = 2^{k+1}$ y tenemos:

$$U(2n) = 2U(n) + 2\left[n + (n-1) + \ldots + (\frac{n}{2} + 1)\right] + 2\left[\sum_{i=1}^{k-1}\left[(\frac{n}{2^i} - 1) + (\frac{n}{2^i} - 2) + \ldots + (\frac{n}{2^{i+1}} + 1)\right]\right]$$

Veamos cual es la justificación de esta ecuación recursiva:

- el sumando $2U(n)$ se debe a la definición inicial del grafo $\hat{G}^{2n}$ a partir de dos subgrafos iguales a $\hat{G}^n$ salvo la numeración de los nodos.
- el sumando $2\left[n + (n-1) + \ldots + (\frac{n}{2} + 1)\right]$ se debe a la adición de n nuevos nodos que cubran respectivamente los intervalos $[i \ldots n]$ para $i = 1 \ldots \frac{n}{2}$, y $[n+1 \ldots j]$ para $j = (\frac{3n}{2} + 1) \ldots 2n$. La adición de un nodo cubriendo un intervalo $[i \ldots n]$ o $[n+1 \ldots j]$ incrementa en 1 el coste de la operación $Update(z, \_)$ para cualquier nodo sin sucesor z perteneciente a dicho intervalo. Cada uno de los sumandos incluídos en el corchete expresa la longitud del intervalo afectado por la adición de un nodo.
- el término $2\left[\sum_{i=1}^{k-1}\left[(\frac{n}{2^i} - 1) + (\frac{n}{2^i} - 2) + \ldots + (\frac{n}{2^{i+1}} + 1)\right]\right]$ responde igualmente a la adición de nuevos nodos al grafo. Cada sumando del corchete mide nuevamente la longitud del intervalo de nodos sin sucesor cubierto por el nuevo nodo. Para todos los nodos del intervalo, el coste de la operación *Update* correspondiente empeora en una unidad. Como vimos en el Teorema 10, es necesario añadir un cierto conjunto de nodos que no estaban presentes en $\hat{G}^n$, especificados en el apartado (1) de los

Lemas 21 y 22. En particular, el Lema 21 afirma que en $\hat{G}^n$ no existen nodos que cubran los intervalos de nodos sin sucesor de la forma $[i \ldots n]$ para los $i \in \left[(n - \frac{n}{2^r} + 2) \ldots (n - \frac{n}{2^{r+1}})\right]$, $r = 1 \ldots k-1$ (considerando $n = 2^{k+1}$), por lo que tales nodos han de ser añadidos en el *Bucle 2* del proceso de `Duplicación`. Por tanto, para un cierto $i_0 = (n - \frac{n}{2^{r_0}} + c)$, se debe incrementar a suma de los costes de las operaciones *Update* en una cantidad de $n - (n - \frac{n}{2^{r_0}} + c) + 1 = \frac{n}{2^{r_0}} - (c - 1)$. El razonamiento es similar para aquellos nodos que se añaden en virtud del Lema 22.

De nuevo la ecuación planteada es exactamente igual a la resuelta en el Teorema 7 para el cálculo del número de unos de una matriz U^n.

Resolviendo la ecuación se obtiene

$$U(n) = \frac{n^2}{2} - \frac{n}{2}\log_2 n + \frac{3n}{2} - 2$$

Finalmente, obtenemos el resultado del teorema:

$$\#Retrieve^n + \#Update^n = \frac{3n^2}{2} - \frac{3n}{2}\log_2 n + \frac{9n}{2} - 2\log_2 n - 4$$

5.3 Equivalencia de Soluciones Matrices-Grafos

En esta sección vamos a demostrar que las parejas de matrices $< R^n, U^n >$ que hemos definido en el apartado 5.1.3, y los grafos $\hat{G}^n$ que hemos definido en el apartado 5.2.2, representan soluciones equivalentes al problema de los rangos variables de dimensión n. Con *soluciones equivalentes* nos referimos a que si interpretamos con un grafo la solución que representa la pareja $< R^n, U^n >$, obtenemos precisamente el grafo $\hat{G}^n$, y si traducimos a una pareja de matrices la solución que define $\hat{G}^n$ obtenemos la pareja $< R^n, U^n >$.

La siguiente observación establece formalmente las condiciones para que un grafo y una pareja de matrices representen soluciones equivalentes.

Observación 15 *Sea una pareja de matrices $< R, U >$ solución para el problema de los rangos variables de dimensión n dentro del modelo matricial, con*

$R = (r_{i,j})_{i=1...n,j=1...m}$, $U = (u_{i,j})_{i=1...m,j=1...n}$, y sea un grafo G solución para el problema de los rangos variables dentro del modelo de grafos. Las matrices y el grafo representan la misma solución si y sólo si

1. $\#Nodos(G) = m$

2. $r_{i,j} = \begin{cases} 1 & si\ y\ sólo\ si \quad j \in Minimal(k,l,G), \quad con \quad i = (l-k+1) + \sum_{s=0}^{k-2}(n-s) \\ 0 & e.o.c \end{cases}$

$$u_{i,j} = \begin{cases} 1 & si\ y\ sólo\ si \quad i \in Antecesores(j,G) \\ 0 & e.o.c \end{cases}$$

A continuación probamos la equivalencia entre las soluciones particulares que hemos propuesto dentro del modelo matricial y dentro del modelo de grafos.

Teorema 12

Sea la pareja de matrices $< R^n, U^n >$ definida como en la Definición 13 del apartado 5.1.3, y sea el grafo $\hat{G}^n$ definido como en la Definición 17 del apartado 5.2.2.

Entonces $< R^n, U^n >$ y $\hat{G}^n$ representan la misma solución al problema de los rangos variables de dimensión n, salvo un posible renombramiento de nodos en $\hat{G}^n$.

Demostración

Lo probamos por inducción sobre n.

1. $n = 2$: el resultado es trivialmente cierto teniendo en cuenta la definición de los algoritmos que implementan las operaciones *Update* y *Retrieve* dentro de cada modelo, el matricial y el de grafo (ver Las Definiciones 7 y 14 respectivamente). En este caso no es necesario renombrar nodos del grafo ni reordenar filas y columnas de $< R^n, U^n >$.

2. $n \Rightarrow 2n$: recordamos que hemos definido $\hat{G}^{2n} = Duplicacion(\hat{G}^n)$, y $< R^{2n}, U^{2n} > = Refinamiento\ (\widehat{R^{2n}}, \widehat{U^{2n}})$, siendo

$$\widehat{R^{2n}} = \begin{pmatrix} R^n & 0 \\ f_n^m(R_1^n(n)) & R_1^n \\ f_n^m(R_2^n(n-1)) & R_1^n \\ f_n^m(R_3^n(n-2)) & R_1^n \\ \vdots & \vdots \\ f_n^m(R_n^n(1)) & R_1^n \\ 0 & R^n \end{pmatrix}, \quad \widehat{U^{2n}} = \begin{pmatrix} U^n & 0 \\ & \\ 0 & U^n \end{pmatrix}$$

Dejamos al lector la comprobación de que la pareja de matrices $\widehat{R^{2n}}$, $\widehat{U^{2n}}$, representa la misma solución que el grafo resultante de la ejecución del *Bucle 1* del código que define el proceso de `Duplicacion` (ver la Definición 16 en la sección 5.2.2).

Posteriormente, cada *paso de ampliación* que se ejecuta en el proceso de *Refinamiento* de las matrices $\widehat{R^{2n}}$, $\widehat{U^{2n}}$, se corresponde con la adición de un nodo en los *bucles 2 y 3* del proceso de `Duplicacion` ■

Ilustramos el teorema anterior generando las soluciones al *problema de los rangos variables* para los tamaños $n = 2$ y $n = 4$.

Ejemplo 4 *Por definición, sabemos que*

$$R^2 \times U^2 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = H^2$$

y el grafo correspondiente $\hat{G}^2$ *es*

Tenemos por tanto

$$\widehat{R^4} = \left(\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ - & - & - & - \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ - & - & - & - \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ - & - & - & - \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{array}\right), \qquad \widehat{U^4} = \left(\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ - & - & - & - \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right)$$

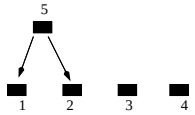
El resultado de la ejecución del Bucle 1 del proceso de `Duplicacion` *aplicado al grafo* $\hat{G}^2$, *es el grafo*

Procedemos a realizar el primer paso de ampliación, asociado al conjunto de columnas $\{1,2\}$ *y filas* $\{2,4,5\}$ *de la matriz* $\widehat{R^4}$ *- marcamos en negrita las posiciones afectadas en* $\widehat{R^4}$ *y las filas concernidas de* $\widehat{U^4}$ *:*

$$\widehat{R^4} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \mathbf{1} & \mathbf{1} & 1 & 0 \\ \mathbf{1} & \mathbf{1} & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \underrightarrow{\textit{nueva columna}} \begin{pmatrix} 1 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 1 & \mathbf{0} & 0 & 0 \\ 0 & 0 & \mathbf{1} & 1 & 0 \\ 0 & 0 & \mathbf{1} & 1 & 1 \\ 0 & 1 & \mathbf{0} & 1 & 0 \\ 0 & 1 & \mathbf{0} & 1 & 1 \\ 0 & 0 & \mathbf{0} & 1 & 0 \\ 0 & 0 & \mathbf{0} & 1 & 1 \\ 0 & 0 & \mathbf{0} & 0 & 1 \end{pmatrix}$$

$$\widehat{U^4} = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{\textit{nueva fila}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

La ejecución de este paso de ampliación sobre las matrices, se corresponde con la adición un nuevo nodo al grafo en el Bucle 2, y los dos arcos correspondientes, quedando



El segundo paso de ampliación afecta a los conjuntos de columnas y filas respectivamente. De nuevo marcamos en negrita las posiciones afectadas en ambas matrices:

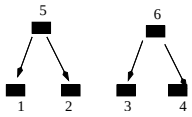
$$\widehat{R^4} \Rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & \mathbf{1} & \mathbf{1} \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & \mathbf{1} & \mathbf{1} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{\textit{nueva columna}} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \mathbf{0} \\ 0 & 0 & 1 & 0 & 0 & \mathbf{0} \\ 0 & 1 & 0 & 0 & 0 & \mathbf{0} \\ 0 & 0 & 1 & 1 & 0 & \mathbf{0} \\ 0 & 0 & 1 & 0 & 0 & \mathbf{1} \\ 0 & 1 & 0 & 1 & 0 & \mathbf{0} \\ 0 & 1 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 1 & 0 & \mathbf{0} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 1 & \mathbf{0} \end{pmatrix}$$

$$\widehat{U^4} \Rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix} \xrightarrow{\textit{nueva fila}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} \end{pmatrix}$$

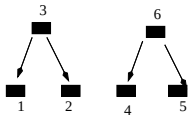
Llamamos $\widehat{R'}^4$, $\widehat{U'}^4$ *a la pareja de matrices resultante de la ejecución los dos pasos de ampliación, y su producto es la matriz* S^4. *La pareja de matrices* $< R^4, U^4 >$ *es precisamente* $< I_\sigma \times \widehat{R'}^4, \widehat{U'}^4 >$, *donde* $I_\sigma = I_{S^4 \to H^4} = I_{3,5} \times I_{4,5}$, *de forma que finalmente se tiene* $R^4 \times U^4 = H^4$:

$$R^4 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \qquad U^4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

Volviendo a la construcción del grafo, la ejecución del segundo paso de ampliación sobre las matrices, se corresponde con la adición de un nuevo nodo y de los dos arcos correspondientes en el Bucle 3, quedando



Y ahora se puede apreciar a qué nos referíamos con lo de que las matrices y el grafo representan la misma solución salvo tal vez renombramiento de nodos del grafo. En efecto, el grafo que se corresponde con la matriz U^4 *es*



(sabemos que el grafo y la matriz U *se determinan unívocamente, pudiendo haber distintas matrices* R *que sirvan como pareja de una* U. *)*

5.4 Comparando Soluciones

En esta sección vamos a comparar las soluciones al problema de los rangos variables definidas en esta tesis en los modelos matricial y de grafos (recordamos que son soluciones equivalentes), con las estructuras de datos solución previamente definidas (ver [19]).

El *Teorema 2* de [19], establece que dado cualquier natural k, existen estructuras de datos solución para el problema de los rangos variables de tamaño n que verifican:

- $\hat{p} = k, \qquad \hat{t} = O(n^{\frac{1}{k-1}}) \qquad (Th\ 2.1)$
- $\hat{t} = k, \qquad \hat{p} = O(n^{\frac{2}{k}}) \qquad (Th\ 2.2)$

siendo $\hat{p}$ la complejidad en el peor caso de la operación *Update* y $\hat{t}$ la complejidad en el peor caso de la operación *Retrieve*.

Demostraremos que nuestras estructuras de datos ofrecen menor complejidad media para el total de las $n+\frac{n(n+1)}{2}$ operaciones, y en el caso de las estructuras propuestas para la demostración de *Th 2.2* requieren un menor número de variables.

Para ello, tendremos efectivamente que calcular la complejidad media y el número de variables correspondientes a tales estructuras, pues ni una ni otra información se deducen directamente del enunciado o la demostración del teorema.

Nos ocupamos en primer lugar de las estructuras de datos propuestas para la demostración del resultado *Th 2.1* , que no son sino árboles con un cierto grado y profundidad.

5.4.1 Complejidad Media de Ciertos Árboles Solución

En [19] se propone la utilización de árboles con n hojas, profundidad k y grado $n^{\frac{1}{k-1}}$ como estructuras de datos solución al problema de los rangos variables de tamaño n, que garantizan $\hat{p} = k, \quad \hat{t} = O(n^{\frac{1}{k-1}})$.

Enunciamos el *problema de los rangos variables* de tamaño n como el diseño de estructuras de datos para el mantenimiento de un vector $A = (a_1 \dots a_n)$ que almacena valores pertenecientes a un semigrupo conmutativo, sobre el cual se quiere realizar operaciones $Retrieve(i,j)$, $Update(j,x)$.

La idea consiste en almacenar en la j-esima hoja del árbol el valor a_j. En cada nodo interno se almacena la suma de los valores almacenados en las hojas del subárbol con raíz en el nodo.

Si se desea realizar una operación $Update(j,x)$, se deberán incrementar en una cantidad x los valores almacenados en los nodos del camino que une la raíz del árbol con la hoja j-esima.

Para ejecutar la operación $Retrieve(i,j)$ se sumará el mínimo conjunto de nodos del árbol que abarque exactamente el intervalo de hojas $[i \dots j]$.

Para calcular la complejidad media que ofrecen estas estructuras de datos, sumamos los costes de las $\frac{n(n+1)}{2}$ operaciones $Retrieve(i,j)$ posibles, y las n posibles operaciones $Update(j, \_)$, donde el coste de una operación se define como el número de nodos del árbol que es necesario sumar en el caso de las operaciones $Retrieve$, o incrementar en el caso de las operaciones $Update$.

En adelante, para facilitar el cálculo, supondremos que el número n de hojas, la profundidad h y el grado r - máximo número de hijos de un nodo - del árbol, verifican la relación $n = r^{h-1}$, es decir, trabajamos con árboles completos en los que todas las hojas están situadas a la misma profundidad.

Obviamente, la suma de los costes de las operaciones $Update(1,\_) \dots Update(n,\_)$ es exactamente hn.

En cuanto a las operaciones $Retrieve(i,j)$, haremos depender de la profundidad del árbol la expresión que nos da la suma de los costes de todas ellas.

Asumimos que la profundidad de la raíz es 1.

La estrategia para obtener el coste total, consiste en calcular el número de operaciones distintas de este tipo en las que aparece involucrado *cada nodo* del árbol, y sumar todas estas cantidades. Evidentemente, para una operación y un nodo concretos, puede ocurrir que, o bien la operación no requiere sumar el contenido del nodo, o bien requiere sumarlo una sola vez.

A continuación probamos un lema que realiza este cálculo para cada uno de los niveles del árbol. Obsérvese que dada la relación que asumimos entre el

grado del árbol y el número de hojas, se deduce que el número de nodos situados a profundidad i es exactamente r^{i-1}.

Lema 23 *Para cualquier profundidad i tal que $1 \leq i \leq h$, se verifica que la suma del número de operaciones Retrieve que afectan a los nodos del árbol situados a dicha profundidad, viene dada por la expresión*

$$S_i^h = \left(\sum_{j=1}^{r^{i-1}} \left[(j-1)r^{h-i} + 1 \right] \left[(r^{i-1} - j)r^{h-i} + 1 \right] \right) - rS_{i-1}^h - r^2 S_{i-2}^h - \ldots - r^{i-1} S_1^h$$

Demostración

El resultado es trivialmente cierto para $i = 1$, con valor $S_1^h = 1$, pues efectivamente la raíz del árbol sólo aparece involucrada en la operación $Retrieve(1, n)$, siendo el único acceso necesario para efectuar esta operación, pues la raíz almacena la suma de los valores contenidos en todas las hojas del árbol.

También es trivial comprobar que para los nodos hoja, el valor viene dado por la expresión

$$S_h^h = \left(\sum_{j=1}^{r^{h-1}} \left[(j-1) + 1 \right] \left[(n-j) + 1 \right] \right) - rS_{h-1}^h - r^2 S_{i-2}^h - \ldots - r^{h-1} S_1^h$$

Llamando R a la suma de los costes de todas las operaciones $Retrieve$, se tiene que $R = \sum_{i=1}^{r^{h-1}} S_i^h$.

A continuación desarrollamos dicha expresión.

$$\begin{aligned} R &= \sum_{i=1}^{r^{h-1}} S_i^h = \sum_{j=1}^{r^{h-1}} j[(n-j)+1] - (r-1)S_{h-1}^h - (r^2-1)S_{i-2}^h - \ldots - (r^{h-1}-1)S_1^h \\ &= \sum_{j=1}^{r^{h-1}} j[(n-j)+1] - (r-1) \sum_{j=1}^{r^{h-2}} [(j-1)r+1][(r^{h-2}-j)r+1] + \\ &\quad (r-1)[rS_{h-2}^h + \ldots + r^{h-2}S_1^h] - (r^2-1)S_{h-2}^h - \ldots - (r^{h-1}-1)S_1^h \\ &= \sum_{j=1}^{r^{h-1}} j[(n-j)+1] - (r-1) \sum_{j=1}^{r^{h-2}} [(j-1)r+1][(r^{h-2}-j)r+1] - \\ &\quad (r-1)S_{h-2}^h - (r^2-1)S_{h-3}^h - \ldots - (r^{h-2}-1)S_1^h \end{aligned}$$

$$
\begin{aligned}
= & \sum_{j=1}^{r^{h-1}} j[(n-j)+1] - (r-1)\sum_{j=1}^{r^{h-2}}[(j-1)r+1][(r^{h-2}-j)r+1] - \\
& (r-1)\sum_{j=1}^{r^{h-3}}[(j-1)r^2+1][(r^{h-3}-j)r^2+1] + (r-1)[rS_{h-3}^h + r^2 S_{h-4}^h + \ldots + r^{h-3}S_1^h] - \\
& (r^2-1)S_{h-3}^h - \ldots - (r^{h-2}-1)S_1^h
\end{aligned}
$$

$$
\begin{aligned}
= & \sum_{j=1}^{r^{h-1}} j[(n-j)+1] - (r-1)\sum_{j=1}^{r^{h-2}}[(j-1)r+1][(r^{h-2}-j)r+1] - \\
& (r-1)\sum_{j=1}^{r^{h-3}}[(j-1)r^2+1][(r^{h-3}-j)r^2+1] - (r-1)S_{h-3}^h - (r^2-1)S_{h-4}^h - \ldots - (r^{h-3}-1)S_1^h
\end{aligned}
$$

Introducimos notación

$$A_l = \sum_{j=1}^{r^{h-l}}[(j-1)r^{l-1}+1][(r^{h-l}-j)r^{l-1}+1]$$

Con esta notación tenemos que

$$
\begin{aligned}
R &= A_1 - (r-1)[A_2+A_3+A_4] - (r-1)S_{h-4}^h - \ldots - (r^{h-4}-1)S_1^h \\
&= A_1 - (r-1)[A_2+A_3+A_4+A_5] - (r-1)S_{h-5}^h - \ldots - (r^{h-5}-1)S_1^h \\
&= A_1 - (r-1)\sum_{l=2}^{h-1} A_l - (r-1)S_1^h \\
&= A_1 - (r-1)\sum_{l=2}^{h} A_l
\end{aligned}
$$

Calculamos en primer lugar A_1 y posteriormente el término general A_l .

$$
\begin{aligned}
A_1 &= \sum_{j=1}^{n} j(n+1-j) = (n+1)\frac{n(n+1)}{2} - \sum_{j=1}^{n} j^2 \\
&= \frac{n(n+1)^2}{2} - \frac{n(n+1)(2n+1)}{6} = (n+1)\frac{3n(n+1)-n(2n+1)}{6} \\
&= n(n+1)\frac{n+2}{6} = \frac{n(n+1)(n+2)}{6}
\end{aligned}
$$

$$A_l = \sum_{j=1}^{r^{h-l}}[(j-1)r^{l-1}+1][(r^{h-l}-j)r^{l-1}+1] = \sum_{j=1}^{r^{h-l}}[(j-1)r^{l-1}+1][n-jr^{l-1}+1]$$

$$
\begin{aligned}
&= \sum_{j=1}^{r^{h-l}} (n+1)(j-1)r^{l-1} + (n+1)r^{h-l} - \sum_{j=1}^{r^{h-l}} r^{l-1} j[(j-1)r^{l-1}+1] \\
&= (n+1)r^{h-l} + (n+1)r^{l-1}\frac{r^{h-l}(r^{h-l}-1)}{2} - r^{2(l-1)}\sum_{j=1}^{r^{h-l}} j(j-1) - r^{l-1}\frac{r^{h-l}(r^{h-l}-1)}{2} \\
&= (n+1)r^{h-l} + \left[(n+1)\frac{n(r^{h-l}-1)}{2}\right] - r^{2(l-1)}\sum_{j=1}^{r^{h-l}} j^2 + r^{2(l-1)}\frac{r^{h-l}(r^{h-l}+1)}{2} - \frac{n(r^{h-l}+1)}{2} \\
&= (n+1)r^{h-l} + \left[(n+1)\frac{n(r^{h-l}-1)}{2}\right] - r^{2(l-1)}\frac{r^{h-l}(r^{h-l}+1)(2r^{h-l}+1)}{6} + \frac{nr^{l-1}(r^{h-l}+1)}{2} - \frac{n(r^{h-l}+1)}{2} \\
&= (n+1)r^{h-l} + \left[(n+1)\frac{n(r^{h-l}-1)}{2}\right] - n\frac{r^{l-1}(r^{h-l}+1)(2r^{h-l}+1)}{6} + \frac{n(r^{l-1}-1)(r^{h-l}+1)}{2} \\
&= (n+1)r^{h-l} + \frac{n(n+1)r^{h-l}}{2} - \frac{n(n+1)}{2} - n\frac{(n+r^{l-1})(2r^{h-l}+1)}{6} + n\frac{n-1-r^{h-l}+r^{l-1}}{2} \\
&= (n+1)r^{h-l} - \frac{n(n+1)}{2} + \frac{n(n-1)}{2} + \frac{n(n+1)r^{h-l}}{2} - \frac{nr^{h-l}}{2} + \frac{nr^{l-1}}{2} - n\frac{2nr^{h-l}+3n+r^{l-1}}{6} \\
&= (n+1)r^{h-l} + \frac{n(-2)}{2} - \frac{n^2}{2} + r^{h-l}\left[\frac{n(n+1)}{2} - \frac{n}{2} - \frac{n^2}{3}\right] + r^{l-1}\left[\frac{n}{2} - \frac{n}{6}\right] \\
&= (n+1)r^{h-l} - (n+\frac{n^2}{2}) + r^{h-l}\left(\frac{n^2}{2} - \frac{n^2}{3}\right) + r^{l-1}\frac{n}{3} \\
&= -(n+\frac{n^2}{2}) + r^{h-l}\frac{n^2}{6} + r^{l-1}\frac{n}{3} + (n+1)r^{h-l} \\
&= \left[n+1+\frac{n^2}{6}\right]r^{h-l} + \frac{n}{3}r^{l-1} - \left(n+\frac{n^2}{2}\right)
\end{aligned}
$$

(la expresión general de los A_l es válida también para el caso $l=1$).

Una vez calculada, retomamos el desarrollo del cálculo de R.

$$R = A_1 - (r-1)\sum_{l=2}^{h} A_l$$

Desarrollamos en primer lugar

$$
\begin{aligned}
\sum_{l=2}^{h} A_l &= \left(n+1+\frac{n^2}{6}\right)\sum_{l=2}^{h} r^{h-l} + \frac{n}{3}\sum_{l=2}^{h} r^{l-1} - (h-1)\left(n+\frac{n^2}{2}\right) \\
&= \left(n+1+\frac{n^2}{6}\right)\sum_{j=0}^{h-2} r^j + \frac{n}{3}\frac{r^h-r}{r-1} - (h-1)\left(n+\frac{n^2}{2}\right)
\end{aligned}
$$

Ahora tenemos

$$R = \frac{n(n+1)(n+2)}{6} - \left(\frac{n^2}{6}+n+1\right)(n-1) - \frac{nr}{3}(n-1) + (r-1)(h-1)\left(n+\frac{n^2}{2}\right)$$

$$
\begin{aligned}
&= \frac{n(n+1)(n+2)}{6} - \frac{n^3}{6} - n^2 - n + \frac{n^2}{6} + n + 1 - \frac{n(n-1)}{3}r + (r-1)(h-1)\left(n + \frac{n^2}{2}\right) \\
&= \frac{n^2}{2} + \frac{n}{3} - n^2 + \frac{n^2}{6} + 1 - \frac{n(n-1)}{3}r + (r-1)(h-1)\left(n + \frac{n^2}{2}\right) \\
&= (r-1)(h-1)\left(n + \frac{n^2}{2}\right) - \frac{n(n-1)}{3}r - \frac{n^2}{3} + \frac{n}{3} + 1 \\
&= (r-1)(h-1)\left(n + \frac{n^2}{2}\right) - \frac{n(n-1)(r+1)}{3} + 1
\end{aligned}
$$

siendo $r = 2, 3 \ldots \sqrt{n}, n$.

Para calcular la suma de los costes de todas las operaciones *Retrieve* y *Update*, recordamos en primer lugar que la suma correspondiente a estas últimas es $U = nh$.

Queremos determinar qué elección de r, h minimiza la expresión

$$
\phi(r) = (r-1)(h-1)\left(n + \frac{n^2}{2}\right) - \frac{n(n-1)(r+1)}{3} + 1 + nh
$$

Considerando que $rh - 1 = n \Rightarrow h = \frac{\log_2 n}{\log_2 r} + 1$, tenemos

$$
\phi(r) = (r-1)\frac{\log_2 n}{\log_2 r}\left(n + \frac{n^2}{2}\right) - \frac{n(n-1)(r+1)}{3} + 1 + n\left(\frac{\log_2 n}{\log_2 r} + 1\right)
$$

Consideramos n constante y r como una variable real, y derivamos la función obteniendo

$$
\begin{aligned}
\phi'(r) &= \frac{\log_2 n}{\log_2 r}\left(n + \frac{n^2}{2}\right) + (r-1)\left(n + \frac{n^2}{2}\right)\log_2 n \frac{-1/r}{(\log_2 r)^2} - \frac{n(n-1)}{3} + n \log_2 n \frac{-1/r}{(log_2 r)^2} \\
&= \log_2 n \left(n + \frac{n^2}{2}\right)\frac{1}{\log_2 r} - \left(n + \frac{n^2}{2}\right)\frac{\log_2 n}{(\log_2 r)^2} + \frac{n^2}{2}\frac{\log_2 n}{r(\log_2 r)^2} - \frac{n(n-1)}{3} \\
&= \log_2 n \left(n + \frac{n^2}{2}\right)\left[\frac{1}{\log_2 r} - \frac{1}{(\log_2 r)^2}\right] + \frac{n^2}{2}\frac{\log_2 n}{r(\log_2 r)^2} - \frac{n(n-1)}{3} \\
&= \log_2 n \left(n + \frac{n^2}{2}\right)\left[\frac{1}{\log_2 r} - \frac{1}{(\log_2 r)^2}\right] + \frac{n^2}{2}\frac{\log_2 n}{r(\log_2 r)^2} - \frac{n^2}{3} + \frac{n}{3} \\
&= \frac{n}{3} + n \log_2 n \left[\frac{1}{\log_2 r} - \frac{1}{(\log_2 r)^2}\right] + \frac{n^2}{2}\log_2 n \left[\frac{1}{\log_2 r} - \frac{1}{(\log_2 r)^2} + \frac{1}{r(\log_2 r)^2}\right] - \frac{n^2}{3}
\end{aligned}
$$

De esta expresión se deduce que $\phi'(r) > 0$ si $r \in [4, \sqrt{n}]$, y por tanto es creciente en dicho intervalo.

Comparando los valores de la función ϕ para 2, 3, 4 y n, llegamos a la conclusión de que si n tiene la forma de potencia de distintos $r_1 \ldots r_s$, es decir $n = r_1^{e_1} = \ldots = r_s^{e_s}$, entonces la mejor elección para el grado del árbol r, es escoger el menor de entre los posibles $r_1 \ldots r_s$.

En particular, en el caso en que n sea una potencia de 2, los árboles binarios ofrecerían una complejidad de

$$\left(n + \frac{n^2}{2}\right) \log_2 n - n(n-1) + 1 + n(\log_2 n + 1).$$

lo que supone una complejidad promedio de $O(\log n)$.

5.4.2 Complejidad Media de Ciertas Estructuras Recursivas

Nos ocupamos ahora de las estructuras de datos propuestas para la demostración del segundo apartado del *Teorema 2* de [19], que garantizan $\hat{t} = k$, $\hat{p} = O(n^{\frac{2}{k}})$, siendo $\hat{p}$ la complejidad en el peor caso de la operación *Update* y $\hat{t}$ la complejidad en el peor caso de la operación *Retrieve*.

Queremos saber cual es la complejidad media para el conjunto de las operaciones que ofrecen dichas estructuras, y también el número de variables requeridas para su implementación.

Recordamos que las soluciones correspondientes a nuestros grafos o matrices, ofrecen una suma de costes del total de operaciones de

$$\frac{3}{2}n^2 - \frac{3}{2}n \log_2 n + \frac{9}{2}n - 2 \log_2 n - 4$$

y requieren un número de variables de

$$n \log n - 2n + 2 \log_2 n + 2$$

Las estructuras que vamos a analizar están definidas recursivamente. Requieren de la definición previa de estructuras de datos que resuelvan el problema más sencillo que hemos llamado *de las sumas parciales*, en el que uno de los extremos del intervalo a sumar en cualquier operación *Retrieve* permanece fijo.

Para garantizar la complejidad mencionada en el teorema, es necesario que estas estructuras de datos para el problema sencillo, se construyan de forma que garanticen a su vez unas ciertas complejidades en el peor caso para los dos tipos de operación. Aquí no estamos interesados en el peor coste de las operaciones, así que omitiremos los detalles referentes a esta cuestión.

Empleamos aquí la notación utilizada en [19] para la definición de ambos tipos de estructuras de datos:

- $D(r,s)$ denota la estructuras de datos solución para el *problema de los rangos variables* donde las operaciones *Retrieve* tienen la forma $Retrieve(i,j)$ tal que $r \leq i \leq j \leq s$. Se entiende que $D(r,s)$ no es más que una traslación de $D(1, s-r+1)$ siendo $s-r+1 = n$.

- $D_L(r,s)$ denota la estructura de datos solución para el *problema de las sumas parciales* donde todas las operaciones *Retrieve* tienen la forma $Retrieve(r,j)$ siendo $r \leq j \leq s$ (es decir, el extremo izquierdo del intervalo está fijo).

- $D_R(r,s)$ denota la estructura de datos solución para el *problema de las sumas parciales* donde todas las operaciones *Retrieve* tienen la forma $Retrieve(i,s)$ siendo $r \leq i \leq s$ (es decir, el extremo derecho del intervalo está fijo).

Las estructuras se definen de forma distinta en función de que la complejidad en el peor caso de la operación *Retrieve* que se pretende obtener, es decir $\hat{t}$, sea par o impar.

Si $\hat{t}$, es par, la estructura $D(1,n)$ se define en función de :

- $D(1, \frac{n}{2})$: esta estructura se utiliza para computar las operaciones $Retrieve(i,j)$ tales que $1 \leq i \leq j \leq \frac{n}{2}$.

- $D(\frac{n}{2}+1, n)$: esta estructura se utiliza para computar las operaciones $Retrieve(i,j)$ tales que $\frac{n}{2}+1 \leq i \leq j \leq n$.

- $D_R(1, \frac{n}{2})$, $D_L(\frac{n}{2}+1, n)$: si los argumentos de la operación $Retrieve(i,j)$ verifican $i \leq \frac{n}{2}$, $\frac{n}{2}+1 \leq j$, entonces se computan $Retrieve(i, \frac{n}{2})$ y

$Retrieve(\frac{n}{2}+1, j)$ en las subestructuras $D_R(1, \frac{n}{2})$ y $D_L(\frac{n}{2}+1, n)$ respectivamente y se suman los resultados obtenidos.

(si $\hat{t}$, es impar la definición es similar pero dividiendo el rango total en un cierto número de intervalos de la misma longitud, en lugar de dividirlo en 2).

Basándonos en esta definición, analizamos la complejidad media que ofrecen estas estructuras de datos. Llamamos $F(n)$ a la suma de las complejidades de las $n + \frac{n(n+1)}{2}$ operaciones *Update* y *Retrieve* posibles asociada a $D(1, n)$.

Tenemos

$$F(2n) = 2F(n) + 2nRet(n) + 2Up(n)$$

donde $Ret(n)$ y $Up(n)$ representan respectivamente la suma de los costes de las n posibles operaciones *Retrieve* y las n posibles operaciones *Update* asociadas tanto a $D_R(1, n)$ como a $D_L(n+1, 2n)$, pues en cuanto a la complejidad de las operaciones ambas subestructuras son obviamente equivalentes.

La justificación para la ecuación anterior, es la siguiente. El valor $Ret(n)$ aparece multiplicado por $2n$ porque la ejecución de las n^2 operaciones $Retrieve(i, j)$ tales que $i \leq n$, $n+1 \leq j$, implica ejecutar n veces cada una de las operaciones $Retrieve(i, n)$ $i = 1 \ldots n$ en $D_R(1, n)$, y otras n veces cada una de las operaciones $Retrieve(n+1, j)$ $j = n+1 \ldots 2n$ en $D_L(n+1, 2n)$. El valor $Up(n)$ aparece multiplicado por 2 porque la ejecución de una operación $Update(j, x)$ implica la actualización de las variables oportunas en las estructuras $D_R(1, n)$ y $D(1, n)$ en caso de que $j \leq n$, o en $D_L(n+1, 2n)$ y $D(n+1, 2n)$ en caso de que $n+1 \leq j$. El sumando $2F(n)$ engloba la suma de todas las operaciones de los dos tipos que afectan a $D(1, n)$ y $D(n+1, 2n)$.

Vemos que necesitamos conocer los valores $Ret(n)$, $Up(n)$, pero nos encontramos con el problema de que las estructuras D_R, D_L se definen en función de las complejidades $\hat{t}$, $\hat{\rho}$ que se quiere obtener ($\hat{t}$ = complejidad peor caso *Retrieves*, $\hat{\rho}$ = complejidad peor caso *Updates*).

Procedemos de la siguiente manera:

1. Calculamos la complejidad media asociada a $D(1, 2n)$ suponiendo que $Ret(n) = n$ (es decir, $\hat{t} = 1$) , lo que implica que $Up(n) = \frac{n(n+1)}{2}$

(recordamos que $Ret(n)$, $Up(n)$ es igual para $D_L(r,s)$ y para $D_R(r,s)$, siendo $n = s-r+1$). Se comprueba que la complejidad media de nuestras soluciones es menor. Incluímos este cálculo porque es el caso que ofrece la menor suma de los costes de las operaciones en este caso, aunque en realidad, dado que estamos suponiendo que $\hat{t} = 1$, deberíamos regirnos por la definición de D(1,n) dada para el caso impar. El cálculo de la complejidad media es sencillo en este caso, pues la solución propuesta es la correspondiente a la pareja de matrices I, H^n de dimensiones respectivas $\frac{n(n+1)}{2} \times \frac{n(n+1)}{2}$ y $\frac{n(n+1)}{2} \times n$. La suma de los costes de todas las operaciones asociada a D(1,n) en este caso, es de $\frac{n(n+1)}{2} + \sum_{j=1}^{n} \frac{j(j+1)}{2} \in \Theta(n^3)$

2. Calculamos la complejidad media asociada a $D(1, 2n)$ suponiendo que $Ret(n) = 2n$ (es decir, el coste medio de las n operaciones *Retrieve* es 2 en las subestructuras auxiliares) , *y desestimando el factor* $Up(n)$ - coste de las operaciones *Update* en las subestructuras - y aún así obtenemos una complejidad media superior a la de nuestras soluciones. De aquí se concluye que para cualquier coste medio de los *Retrieve* en D_L, D_R que sea mayor que 2, las soluciones definidas por nuestros grafos tienen menor complejidad media.

Procedemos a demostrar los resultados en el orden que acabamos de fijar.

Lema 24

Sea $D(1, n, k = 1)$ *la estructura de datos solución descrita arriba (ver el segundo apartado del Teorema 2 de [19]) donde el tercer argumento denota que las subestructuras auxiliares utilizadas garantizan una complejidad media de* $k = 1$ *para las operaciones Retrieve. En ese caso, la complejidad media del conjunto de las operaciones asociada a* $D(1, n, k)$ *es de* $\frac{3}{2}n^2 + \frac{n}{2}\log_2 n + \frac{n}{2}$.

Demostración

El hecho de que la complejidad media de las operaciones *Retrieve* ofrecida por cualquier estructura de datos solución al *problema de las sumas parciales* de tamaño n sea 1, implica que la suma de los costes de las n operaciones *Retrieve* es exactamente n y que la suma de los costes de las n operaciones

Update es exactamente $\frac{n(n+1)}{2}$. Esto último puede resultar más fácil de intuir si pensamos en la representación matricial que tendrían estas estructuras de datos dentro del modelo algebraico propuesto por M.L.Fredman(ver apartado 2.3.1 en el Capítulo 2), y la única posibilidad es que la matriz asociada a las operaciones *Retrieve* sea la identidad y la correspondiente a los *Update* la matriz T.

Por tanto, en la ecuación que expresa la suma de las complejidades de todas las operaciones asociada a la estructura $D(1,2n)$, debemos sustituir $Ret(n)$ por n y $Up(n)$ por $\frac{n(n+1)}{2}$, obteniendo

$$\begin{aligned} F(2n) &= 2F(n) + 2nRet(n) + 2Up(n) = 2F(n) + 2n^2 + n^2 + n \\ &= 2F(n) + 3n^2 + n = 2^2F(\frac{n}{2}) + \frac{3n^2}{2} + 3n^2 + 2n \\ &= \ \ldots \ = 2^{k+1}F(\frac{n}{2^k}) + 3n^2(1 + \frac{1}{2} + \frac{1}{4} + \ldots + \frac{1}{2^k}) + (k+1)n \end{aligned}$$

Para $k = \log_2 n$ y teniendo en cuenta que $F(1) = 2$, tenemos por tanto que

$$\begin{aligned} F(2n) &= 4n + 3n^2(2 - \frac{1}{n}) + n\log_2 n + n = 2n + 6n^2 + n\log_2 n \Rightarrow \\ F(n) &= \frac{3}{2}n^2 + \frac{n}{2}\log_2 n + \frac{n}{2} \end{aligned}$$

Nuestra solución ofrece una suma de los costes ligeramente mejor, exactamente de

$$\frac{3}{2}n^2 - \frac{3}{2}n\log_2 n + \frac{9}{2}n - 2\log_2 n - 4 \ \blacksquare$$

Lema 25

Sea $D(1,n,k=2)$ la estructura de datos solución definida en la demostración del segundo apartado del Teorema 2 de [19], donde el parámetro k denota que las subestructuras auxiliares utilizadas garantizan una complejidad media de $k=2$ para las operaciones Retrieve. Sin tener en cuenta el coste medio de las operaciones Update en las subestructuras auxiliares, obtenemos que la complejidad media del conjunto de las operaciones asociada a $D(1,n,k)$ es mayor que $2n^2 - \frac{3}{2}n$.

Demostración

En este caso, la ecuación recursiva es

$$F(2n) \;=\; 2F(n) + 2nRet(n) + 2Up(n) \;>\; 2F(n) + 4n^2$$

es decir, hemos sustituído $Ret(n)$ por $2n$ y hemos desestimado el factor $2Up(n)$.

Desarrollando esta sencilla ecuación se obtiene

$$F(2n) \;>\; 8n^2 - 3n \;\Rightarrow\; F(n) \;>\; 2n^2 - \frac{3}{2}n \;\blacksquare$$

Observación 16 *Obsérvese que del lema anterior se deduce inmediatamente que para cualquier coste medio de los Retrieve en D_L, D_R que sea mayor que 2, las soluciones definidas por nuestros grafos tienen menor complejidad media (al menos en el caso en que $\hat{t}$ tenga valor par, pues dejamos la comprobación del caso impar al lector interesado) .*

Número de Variables

El número de variables $V(n)$ de la estructura $D(1,n)$ (en general $D(r,s)$ tal que $s - r + 1 = n$), se rige por la ecuación recursiva

$$V(n) \geq 2V(\frac{n}{2}) + 2\frac{n}{2}$$

tomando como caso base de la recursión el valor $V(1) = 1$.

La ecuación refleja el hecho de que el número mínimo de variables que requiere cada una de las subestructuras auxiliares utilizadas $D_R(1, \frac{n}{2}$, $D_L(\frac{n}{2}+1, n)$ requiere un mínimo de $\frac{n}{2}$ variables (ver Lema 1 en el Capítulo 3 de esta tesis).

Desarrollamos la ecuación,

$$\begin{aligned} V(n) \;&\geq\; 2V(\frac{n}{2}) + n = 2[2V(\frac{n}{2^2}) + 2\frac{n}{2^2}] + n \\ &=\; 2^2V(\frac{n}{2^2}) + 2n = 2^2[2V(\frac{n}{2^3}) + 2\frac{n}{2^3}] + 2n \\ &=\; 2^3V(\frac{n}{2^3}) + 3n = \ldots = 2^kV(\frac{n}{2^k}) + kn \end{aligned}$$

Tomando $\frac{n}{2^k} = 1 \Rightarrow k = \log_2 n$, se obtiene finalmente

$$V(n) \geq n\log_2 n + n \;\blacksquare$$

Capítulo 6

Nueva Formulación del Problema de las Sumas Parciales en el Modelo de Grupo

El estudio de las cotas inferiores para la complejidad de las operaciones de acceso y modificación sobre un vector V, es más difícil en el caso en el que se asume que el vector V almacena valores pertenecientes a un *grupo* conmutativo cualquiera, en lugar de a un *semigrupo*. En el caso *semigrupo* se conoce la cota inferior óptima para la complejidad media de las operaciones (ver [15]), y tambien se han definido estructuras de datos optimales respecto a la complejidad en el peor caso (ver [14]).

En el caso de *grupo*, por el contrario, en el que la resta está permitida, sigue existiendo una distancia entre la mejor cota superior y la mejor cota inferior tanto para la complejidad media de las operaciones como para la complejidad en el peor caso (ver [14]).

Proponemos una manera diferente de abordar el problema. Consiste en replantearlo en términos de la forma de Jordan J correspondiente a la matriz

T. Para ello necesitamos calcular la matriz Q del cambio de base tal que si $R \times U = T$, entonces

$$Q^{-1} \times R \times U \times Q = Q^{-1} \times T \times Q = J.$$

Una curiosidad respecto a la matriz Q que obtendremos es que, asumiendo que trabajamos en $\mathbf{Z_2}$, la matriz Q resulta ser practicamente igual al triángulo de Pascal - módulo 2 - además de que resulta ser igual a su inversa.

6.1 Matriz del Cambio de Base y Forma de Jordan de la Matriz T

Antes de poder definir la matriz de paso Q y la forma de Jordan J correspondientes a T, debemos hacer unos cálculos previos.

Inicialmente calculamos la forma de Jordan y la matriz de paso de T suponiendo que trabajamos en los enteros, y una vez obtenida esta matriz de paso, aplicaremos la operación módulo a todas las componentes de la matriz obteniendo la Q que buscamos.

6.1.1 Un T-Generador de $\mathbf{R^n}$

Comenzamos demostrando que el vector

$$e_1 = (1, 0 \ldots 0)$$

es un T-generador de R^n. Probarlo es equivalente a probar que $e_1, T(e_1), T^2(e_1), \ldots, T^{n-1}(e_1)$ son linealmente independientes.

Veamos como son dichos vectores:

$$\begin{aligned} e_1 &= (1, 0, \ldots, 0) \\ T(e_1) &= (1, 1, \ldots, 1) \\ T^2(e_1) &= (1, 2, 3, 4, \ldots) \\ &\ldots \end{aligned}$$

y en general,

$$T^k(e_1)_j = \sum_{z=1}^{j} T^{k-1}(e_1)_z$$

.

Para ver que los vectores son linealmente independientes, comprobaremos que

$$D_n = \begin{vmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 3 & \cdots & n \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{vmatrix} \neq 0$$

Llamando $T^k(e_1)_i = a_i^k$, tenemos que

$$D_n = \begin{vmatrix} a_1^0 & \cdots & a_n^0 \\ a_1^1 & \cdots & a_n^1 \\ \vdots & \ddots & \vdots \\ a_1^{n-1} & \cdots & a_n^{n-1} \end{vmatrix}$$

$$\begin{aligned} a_1^0 &= 1 \\ a_i^0 &= 0 \quad \forall\, i > 1 \\ a_i^k &= \textstyle\sum_{j=1}^{i} a_j^{k-1} \quad \forall\, k \geq 1 \end{aligned}$$

Vemos, igualmente, que

$$a_i^k - a_{i-1}^k = a_i^{k-1}, \quad \forall\, k > 0$$

o lo que es lo mismo

$$a_i^k - a_i^{k-1} = a_{i-1}^k, \quad \forall\, k > 0$$

A continuacion, describimos un proceso que se organiza en $n-1$ etapas, en cada una de las cuales realizamos una serie de combinaciones lineales de las columnas del determinante que no afectan a su valor.

- **etapa 1:** restamos a cada columna la columna anterior, con excepción de las dos primeras columnas que no se modifican. Es decir, para todo $i = 1, \ldots, (n-2)$, se resta la columna $(n-i)$ de la columna $(n-i+1)$.

  Tras el proceso, tenemos una nueva matriz

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 1 & a_3^1 - a_2^1 & a_4^1 - a_3^1 & \cdots & a_n^1 - a_{n-1}^1 \\ 1 & 2 & a_3^2 - a_2^2 & a_4^2 - a_3^2 & \cdots & a_n^2 - a_{n-1}^2 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & (n-1) & a_3^{n-1} - a_2^{n-1} & a_4^{n-1} - a_3^{n-1} & \cdots & a_n^{n-1} - a_{n-1}^{n-1} \end{pmatrix}$$

con el mismo determinante.

Expandiendo el determinante por la primera fila, vemos que su valor es el valor de su menor

$$D_{n-1} = \begin{vmatrix} 1 & a_3^1 - a_2^1 & \cdots & a_n^1 - a_{n-1}^1 \\ 2 & a_3^2 - a_2^2 & \cdots & a_n^2 - a_{n-1}^2 \\ \cdots & & & \\ (n-1) & a_3^{n-1} - a_2^{n-1} & \cdots & a_n^{n-1} - a_{n-1}^{n-1} \end{vmatrix}$$

Dado que $a_i^k - a_{i-1}^k = a_i^{k-1}$, tenemos

$$D_{n-1} = \begin{vmatrix} 1 & a_3^0 & a_4^0 & \cdots & a_n^0 \\ 2 & a_3^1 & a_4^1 & \cdots & a_n^1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ (n-1) & a_3^{n-2} & a_4^{n-2} & \cdots & a_n^{n-2} \end{vmatrix} .$$

- **etapa 2:** operamos sucesivamente las filas - en el paso anterior se trataba de las columnas - de la matriz correspondiente al determinante D_{n-1} (dimensión $(n-1)\times(n-1)$) de forma que a cada fila se le resta la anterior, dejando la primera fila como estaba. Es decir, para todo $i = 2, \ldots, (n-1)$, restamos a la fila $(n-i+1)$ la fila $(n-i)$.

  Obtenemos una nueva matriz cuyo determinante tiene el mismo valor:

$$\begin{vmatrix} 1 & a_3^0 & a_4^0 & \cdots & a_n^0 \\ 1 & a_3^1 - a_3^0 & a_4^1 - a_4^0 & \cdots & a_n^1 - a_n^0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & a_3^{n-2} - a_3^{n-3} & a_4^{n-2} - a_4^{n-3} & \cdots & a_n^{n-2} - a_n^{n-3} \end{vmatrix}$$

Aplicamos ahora que $a_i^k - a_i^{k-1} = a_{i-1}^k$, y el determinante de la matriz nos queda

$$D_{n-1}^1 = \begin{vmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & a_2^1 & a_3^1 & \cdots & a_{n-1}^1 \\ 1 & a_2^2 & a_3^2 & \cdots & a_{n-1}^2 \\ \cdots & & & & \\ 1 & a_2^{n-2} & a_3^{n-2} & \cdots & a_{n-1}^{n-2} \end{vmatrix}$$

Expandiendo por la primera fila, obtenemos

$$D_{n-2} = \begin{vmatrix} a_2^1 & a_3^1 & \cdots & a_{n-1}^1 \\ a_2^2 & a_3^2 & \cdots & a_{n-1}^2 \\ \vdots & \ddots & & \vdots \\ a_2^{n-2} & a_3^{n-2} & \cdots & a_{n-1}^{n-2} \end{vmatrix}$$

- **etapa 3:** en esta etapa operaremos de nuevo las columnas de la matriz correspondiente al determinante resultado de la etapa anterior, de dimensión $(n-2)$. Hay que restar a cada columna la anterior, dejando la primera columna igual, aplicar que $a_i^k - a_{i-1}^k = a_i^{k-1}$ y expandir por la primera fila.

...

- **etapa k-esima :** operamos las columnas (o filas, según k sea impar o par respectivamente) de la matriz correspondiente a D_{n-k+1}, restando a cada columna (fila) la inmediatamente anterior, permaneciendo la primera de ellas inalterada. Se obtiene una nueva matriz con el mismo determinante, se aplica que $a_i^k - a_{i-1}^k = a_i^{k-1}$, y se expande el determinante por la primera fila obteniendo D_{n-k}

Tras las $n-1$ etapas anteriores, se concluye que el valor del determinante es 1, con lo que se demuestra que los vectores $e_1, T(e_1), T^2(e_1), \ldots, T^{n-1}(e_1)$ son linealmente independientes.

La matriz T sobre el cuerpo $\mathbf{R}$ representa un endomorfismo del espacio vectorial $\mathbf{R^n}$ en $\mathbf{R^n}$.

Un teorema conocido de álgebra lineal afirma que dados un espacio vectorial cualquiera E de dimensión finita y un endomorfismo f definido sobre él, podemos descomponer E en una suma de subespacios vectoriales *f-cíclicos* y *f-irreducibles*, distintos todos ellos del subespacio vectorial $\{0\}$. Por tanto, podemos expresar $\mathbf{R^n}$ como

$$\mathbf{R^n} = \mathbf{L_1} \oplus \mathbf{L_2} \oplus \ldots \oplus \mathbf{L_r}$$

con $\mathbf{L_i} \neq \{0\} \quad \forall\ i = 1 \ldots r,$ y siendo $\mathbf{L_i}$ T-cíclico y T-irreducible para todo $i = 1 \ldots r,$ es decir, para todo $\mathbf{L_i}$ existe un vector $a_i \in \mathbf{R^n}$ tal que $\mathbf{L_i} = L(a_i, T(a_i), T^2(a_i) \ldots)$. Decimos que a_i es un T-generador de $\mathbf{L_i}$.

Observación 17

En particular, hemos demostrado que el propio $\mathbf{R^n}$ *es T-cíclico y T-irreducible, y que* e_1 *es un T-generador para* $\mathbf{R^n}$.

6.1.2 Obtención de una Base de Vectores y Matriz J

Para todo $i = 0 \ldots (n-2)$ se tiene que $T(T^i(e_1)) = T^{i+1}(e_1),$ y obviamente también es cierto $T(T^{n-1}(e_1)) = T^n(e_1)$. Este último vector es necesariamente una combinación lineal de $e_1, T(e_1) \ldots T^{n-1}(e_1)$, por lo que podemos escribir

$$T^n(e_1) = -\gamma_{0,n} e_1 - \gamma_{1,n} T(e_1) - \ldots - \gamma_{n-1,n} T^{n-1}(e_1)$$

es decir,

$$\sum_{k=0}^{n} \gamma_{k,n} T^k(e_1) = 0 \qquad \text{con} \qquad \gamma_{n,n} = 1$$

Por tanto, respecto a la base formada por los vectores

$$e_1, T(e_1), T^2(e_1), T^3(e_1), \ldots, T^{n-1}(e_1)\ ,$$

la matriz T tiene la forma

$$C = \begin{pmatrix} 0 & 0 & 0 & \cdots & -\gamma_{0,n} \\ 1 & 0 & 0 & \cdots & -\gamma_{1,n} \\ 0 & 1 & 0 & \cdots & -\gamma_{2,n} \\ 0 & 0 & 1 & \cdots & -\gamma_{3,n} \\ \cdots & & & & \\ 0 & 0 & 0 & \cdots & -\gamma_{n-1,n} \end{pmatrix}$$

El polinimio característico de C es igual al polinomio característico de T, dado que es un invariante que no depende de la base que tomemos en el espacio vectorial.

El polinomio característico de la matriz T es el determinante $|T - \lambda I| = (1 - \lambda)^n$. Los únicos divisores de este polinomio deberán por tanto tener la forma $(1 - \lambda)^k$.

Por otra parte, un resultado de álgebra lineal establece que el *polinomio minimal* de un endomorfismo divide al polinomio característico, y por tanto el polinomio minimal tendrá necesariamente la forma $(T - I)^k$ para un cierto $k \leq n$. Pero obviamente, $(T - I)^k$ sólo es el endomorfismo 0 para $k = n$. En conclusión, el polinomio minimal es el propio $(T - I)^n$.

Ahora, dado que e_1 es un *T-generador* de $\mathbf{R^n}$, podemos concluir que los vectores

$$e_1,\ (T - I)e_1,\ (T - I)^2(e_1),\ \ldots,\ (T - I)^{n-1}(e_1)$$

son linealmente independientes; en otro caso, el grado del *T-anulador* de e_1 sería menor o igual que n, pues un resultado conocido afirma que el *T-anulador* de un vector del espacio vectorial, debe dividir al polinomio minimal. Particularizando para el vector e_1, tendremos que su *T-anulador* tendrá por tanto la forma $(T - I)^k$ para un cierto $k \leq n$.

Un teorema algebraico afirma que si un vector a es un *f-generador* de un espacio vectorial E f-cíclico, donde f es un endomorfismo sobre E, entonces la dimensión de E es igual al grado del *f-anulador* de a.

En cualquier caso, y sin recurrir a este último teorema, es trivialmente cierto que para anular el vector $e_1 = (1, 0, 0 \ldots 0)$ el grado ha de ser necesariamente $k = n$.

Por tanto los vectores

$$e_1,\ (T - I)e_1,\ (T - I)^2(e_1),\ \ldots,\ (T - I)^{n-1}(e_1)$$

forman una base para $\mathbf{R^n}$.

Respecto a esta base de vectores, la matriz T tiene la forma

$$J = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 \\ \cdots & & & & & \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix}$$

ya que $T\left[(T-I)^i(e_1)\right] = (T-I)^{i+1}(e_1) + (T-I)^i(e_1)$ (esto último es porque $(T-I)^{i+1}(e_1) + (T-I)^i(e_1) = [(T-I)+I]\,(T-I)^i = T((T-I)^i)$).

6.1.3 Matriz Q del Cambio de Base

Veamos como son los vectores $e_1,\ (T-I)e_1 \ldots (T-I)^{n-1}(e_1)$:

$$e_1 = (\,1 \quad 0 \quad \cdots \quad 0\,)$$

$$(T-I)e_1 = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 1 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 1 & 1 & \cdots & 0 & 0 \\ \cdots & & & & & & \\ 1 & 1 & 1 & 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

$$(T-I)^2e_1 = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 1 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 1 & 1 & \cdots & 0 & 0 \\ \cdots & & & & & & \\ 1 & 1 & 1 & 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 2 \\ 3 \\ \vdots \\ (n-2) \end{pmatrix}$$

$$(T-I)^3 e_1 = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 1 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 1 & 1 & \cdots & 0 & 0 \\ \cdots & & & & & & \\ 1 & 1 & 1 & 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 2 \\ 3 \\ \vdots \\ (n-2) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 3 \\ \vdots \\ \sum_{i=1}^{n-3} i \end{pmatrix}$$

En general, si llamamos

$$(T-I)^k e_1 = \begin{pmatrix} a_1^k \\ a_2^k \\ \vdots \\ a_n^k \end{pmatrix} \quad \forall\ k > 1$$

tenemos

$$\begin{aligned} ((T-I)^k e_1)_j &= 0 && \forall\ j = 1, \ldots, k \\ ((T-I)^k e_1)_{k+1} &= 1 \\ ((T-I)^k e_1)_j &= \textstyle\sum_{i=1}^{j-1} a_i^{k-1} \end{aligned}$$

¿Cuales son entonces las matrices Q y Q^{-1} tales que $J = Q^{-1}TQ$?

Q viene dada por:

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 3 & 3 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 4 & \cdots & 0 & 0 & 0 \\ \cdots & & & & & & & & \\ & & & \cdots & \cdots & & 1 & 0 & 0 \\ 0 & 1 & (n-3) & & & \cdots & (n-3) & 1 & 0 \\ 0 & 1 & (n-2) & & & \cdots & & (n-2) & 1 \end{pmatrix}$$

y afirmamos que Q^{-1} es la matriz:

$$Q^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 3 & -3 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 1 & -4 & 6 & -4 & \cdots & 0 & 0 & 0 \\ \cdots & & & & & & & & \\ \cdots & & & \cdots & \cdots & & 1 & 0 & 0 \\ \cdots & & & & & \cdots & -(n-3) & 1 & 0 \\ \cdots & & & & & \cdots & & -(n-2) & 1 \end{pmatrix}$$

Para probarlo veremos que si llamamos

$$Q = (a_{i,j})_{i,j=1\ldots n}$$

$$Q^{-1} = (b_{i,j})_{i,j=1\ldots n}$$

entonces se verifica

$$b_{i,j} = (-1)^{i+j} a_{i,j} \qquad [1]$$

Procedemos a comprobarlo. Para ello, observamos la matriz Q, y vemos que

$$a_{i,j} = \binom{i-2}{j-2} \quad \forall\ j = 2 \ldots i$$

y también

$$a_{i,2+k} = a_{i,i-k} \qquad \forall\ k = 0 \ldots (i-2)$$

De momento asumiremos - lo probaremos más tarde - que la matriz Q^{-1} se puede expresar como:

$$Q^{-1} = \begin{pmatrix} Q_{n-1}^{-1} & \overline{\mathbf{v}} \\ \mathbf{w}\mathbf{Q}_{\mathbf{n-1}}^{-\mathbf{1}} & 1 \end{pmatrix}$$

siendo

$$\overline{\mathbf{v}} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

y

$$\mathbf{w} = -\begin{pmatrix} a_{n,1} & a_{n,2} & \cdots & a_{n,n-1} \end{pmatrix}$$

Probaremos que $b_{i,j} = (-1)^{i+j} a_{i,j}$ por inducción, así que supondremos que es cierto para las $(n-1)$ filas y columnas primeras - que se corresponden con Q_{n-1}^{-1} - y probaremos que es cierto tambien para la última fila - es trivialmente cierto para la última columna.

Dada la forma que tiene la matriz Q^{-1}, tenemos que

$$\begin{aligned} b_{n,j} &= -\begin{pmatrix} a_{n,1} & a_{n,2} & \cdots & a_{n,n-1} \end{pmatrix} \begin{pmatrix} b_{1,j} \\ \vdots \\ b_{n-1,j} \end{pmatrix} \\ &= -\begin{pmatrix} a_{n,1}(-1)^{j+1} a_{1,j} + & \cdots & +(-1)^{j+(n-1)} a_{n,n-1} a_{n-1,j} \end{pmatrix} \end{aligned}$$

Recordamos que $Q = (a_{i,j})$ es una matriz triangular, y por tanto

$$a_{1,j} = a_{2,j} = \ldots = a_{j-1,j}$$

y tenemos

$$\begin{aligned} b_{n,j} &= -\begin{pmatrix} a_{n,j} a_{j,j} + & \cdots & +(-1)^{j+(n-1)} a_{n,n-1} a_{n-1,j} \end{pmatrix} \\ &= \begin{pmatrix} -a_{n,j} a_{j,j} + & a_{n,j+1} a_{j+1,j} - & \cdots & +(-1)^{j+n} a_{n,n-1} a_{n-1,j} \end{pmatrix} \end{aligned}$$

Dado que

$$a_{i,j} = \binom{i-2}{j-2} \quad \forall\, j = 2 \ldots i \ ,$$

tenemos que probar por tanto

$$\begin{aligned} b_{n,j} &= (-1)^{n+j} \binom{n-2}{j-2} \\ &= -\binom{n-2}{j-2}\binom{j-2}{j-2} + \binom{n-2}{j-1}\binom{j-1}{j-2} + \ldots + (-1)^{n+j}\binom{n-2}{n-3}\binom{n-3}{j-2} \end{aligned}$$

Sea $m = n-2$, $k = j-2$. Hemos de demostrar

$$(-1)^{k+m}\binom{m}{k} = -\binom{m}{k}\binom{k}{k} + \binom{m}{k+1}\binom{k+1}{k} \ldots + (-1)^{k+m}\binom{m}{m-1}\binom{m-1}{k}$$

Aplicamos que

$$\binom{m}{k} = \frac{m!}{k!\,(m-k)!}$$

y observamos que el cociente $(\frac{m!}{k!})$ está presente en todos los términos del lado derecho de la igualdad, por lo que suprimimos dicho cociente. Ahora todos los términos del lado derecho se ajustan a la expresión

$$\begin{aligned}(-1)^{j+1}\binom{m}{k+j}\binom{k+j}{k} &= (-1)^{j+1}\frac{m!\,(k+j)!}{(k+j)!\,[(m-(k+j)]!\,k!\,j!} \\ &= (-1)^{j+1}\frac{m!}{[m-(k+j)]!\,k!\,j!} \quad \forall\ j = 0\ldots(m-k-1)\end{aligned}$$

Estamos tratando de probar

$$(-1)^{k+m}\frac{1}{(m-k)!} = -\frac{1}{(m-k)!\,0!}+\ldots+(-1)^{j+1}\frac{1}{(m-(k+j))!\,j!}+\ldots+(-1)^{m-k}\frac{1}{1!\,(m-1-k)!}$$

lo que es equivalente a probar

$$(-1)^{k+m} = -\binom{m-k}{m-k}+\binom{m-k}{m-k-1}+\ldots+(-1)^{j+1}\binom{m-k}{m-k-j}+\ldots+(-1)^{k+m}\binom{m-k}{1}$$

o lo que es lo mismo

$$\begin{aligned}0 &= -\binom{m-k}{m-k}+\binom{m-k}{m-k-1}+\ldots+(-1)^{m-k}\binom{m-k}{1}+(-1)^{k+m+1}\binom{m-k}{0} \\ &= -\left[\binom{m-k}{m-k}-\binom{m-k}{m-k-1}+\ldots+(-1)^{m-k-1}\binom{m-k}{1}+(-1)^{m-k}\binom{m-k}{0}\right]+ \\ &\quad +(-1)^{m-k}\binom{m-k}{0}+(-1)^{k+m+1}\end{aligned}$$

Pero

$$\begin{aligned}(-1)^{m-k}+(-1)^{k+m+1} &= (-1)^{m-k}(-1)^{2k}+(-1)^{m+k+1} \\ &= (-1)^{m+k}-(-1)^{m+k} \\ &= 0\,,\end{aligned}$$

y con esto hemos completado la demostración de que $b_{i,j} = (-1)^{i+j}a_{i,j}$.

Nos queda por demostrar que la matriz Q^{-1} se puede expresar como

$$Q^{-1} = \begin{pmatrix} Q_{n-1}^{-1} & \overline{\mathbf{v}} \\ \mathbf{w}\mathbf{Q}_{\mathbf{n-1}}^{-1} & 1 \end{pmatrix}$$

siendo

$$\overline{\mathbf{v}} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

y

$$\mathbf{w} = -\begin{pmatrix} a_{n,1} & a_{n,2} & \cdots & a_{n,n-1} \end{pmatrix}$$

Sea

$$J = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 \\ \cdots & & & & & \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix}$$

$$T = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 0 \\ \cdots & & & & \\ 1 & 1 & 1 & \cdots & 1 \end{pmatrix}$$

Sabemos que Q es triangular y que puede expresarse como

$$Q = \begin{pmatrix} Q_{n-1} & \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ \alpha & 1 \end{pmatrix}$$

siendo

$$\alpha = \begin{pmatrix} a_{n,1} & a_{n,2} & \cdots & a_{n,n-1} \end{pmatrix}$$

La matriz Q^{-1} tiene que verificar $Q \times Q^{-1} = I$, así que podemos escribir :

$$\begin{pmatrix} Q_{n-1} & \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ \alpha & 1 \end{pmatrix} \times \begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} I_{n-1} & \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 0 & \cdots & 0 \end{pmatrix} & 1 \end{pmatrix}$$

De aquí podemos deducir:

- $Q_{n-1} \times A + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \times C = I_{n-1}$ siendo I_{n-1} la matriz identidad de dimensión $(n-1)$, y por tanto $A = Q_{n-1}^{-1}$.

- $Q_{n-1} \times B + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \times D = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, por lo que $Q_{n-1} \times B = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$.

  Sabemos que $det(Q_{n-1}) \neq 0$, así que $Ker(Q_{n-1}) = \{0\}$, y en conclusión, $B = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$.

- $(a_{n,1} \quad a_{n,2} \quad \cdots \quad a_{n,n-1}) \times A + C = (0 \quad 0 \quad \cdots \quad 0)$, y por tanto,

  $C = -((a_{n,1} \quad a_{n,2} \quad \cdots \quad a_{n,n-1})) \times A = -((a_{n,1} \quad a_{n,2} \quad \cdots \quad a_{n,n-1})) \times Q_{n-1}^{-1}$

- $(a_{n,1} \quad a_{n,2} \quad \cdots \quad a_{n,n-1}) \times \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + D = 1$, lo que nos permite concluir que $D = (1)$.

Una vez demostrado que Q^{-1} se puede expresar como

$$Q^{-1} = \begin{pmatrix} Q_{n-1}^{-1} & \overline{\mathbf{v}} \\ \mathbf{w}\mathbf{Q}_{\mathbf{n-1}}^{-\mathbf{1}} & 1 \end{pmatrix}$$

podemos comprobar que efectivamente

$$T = Q \times J \times Q^{-1}.$$

Llamamos T_n a la matriz T de dimensión n.

Podemos descomponer la matriz J de forma que tenemos que probar

$$T_n = Q \times (I + \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}) \times Q^{-1}$$

Podemos descomponer igualmente T_n como

$$T_n = I + \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ \cdots & & & & & \\ 1 & 1 & 1 & \cdots & 1 & 0 \end{pmatrix}$$

de forma que debemos probar

$$I + \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ \cdots & & & & & \\ 1 & 1 & 1 & \cdots & 1 & 0 \end{pmatrix} = I + (Q \times \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \times Q^{-1})$$

es decir

$$\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ \cdots & & & & & \\ 1 & 1 & 1 & \cdots & 1 & 0 \end{pmatrix} = Q \times \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \times Q^{-1}$$

Llamamos

$$B_n = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}$$

y tenemos

$$
\begin{aligned}
Q \times B_n &= \begin{pmatrix} Q_{n-1} & \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ \alpha & 1 \end{pmatrix} \times \begin{pmatrix} B_{n-1} & \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ (0 \quad 0 \quad \cdots \quad 0 \quad 1) & 0 \end{pmatrix} \\
&= \begin{pmatrix} Q_{n-1}B_{n-1} & \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ \alpha B_{n-1} + (0 \quad 0 \quad \cdots \quad 0 \quad 1) & 0 \end{pmatrix}
\end{aligned}
$$

siendo $\alpha = (\, a_{n,1} \quad a_{n,2} \quad \cdots \quad a_{n,n-1} \,)$.

Por tanto

$$
\begin{aligned}
Q \times B_n \times Q^{-1} &= \begin{pmatrix} Q_{n-1}B_{n-1} & \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ \alpha B_{n-1} + (0 \quad 0 \quad \cdots \quad 0 \quad 1) & 0 \end{pmatrix} \times \begin{pmatrix} Q_{n-1}^{-1} & \overline{\mathbf{v}} \\ \mathbf{w}\mathbf{Q}_{\mathbf{n-1}}^{-1} & 1 \end{pmatrix} \\
&= \begin{pmatrix} Q_{n-1}B_{n-1}Q_{n-1}^{-1} & \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\ \alpha B_{n-1}Q_{n-1}^{-1} + (0 \quad 0 \quad \cdots \quad 0 \quad 1)\, Q_{n-1}^{-1} & \end{pmatrix}
\end{aligned}
$$

Por hipótesis de inducción sabemos que

$$Q_{n-1} \times B_{n-1} \times Q_{n-1}^{-1} = T_{n-1} - I \ ,$$

así que sólo tenemos que probar

$$\alpha \times B_{n-1} \times Q_{n-1}^{-1} + (0 \quad 0 \quad \cdots \quad 0 \quad 1) \times Q_{n-1}^{-1} = (1 \quad 1 \quad \cdots \quad 1)$$

Desarrollamos el lado derecho de la igualdad,

$$
\begin{aligned}
[\alpha \times B_{n-1} + (0 \ldots 0\ 1)]Q_{n-1}^{-1} &= [(\, a_{n,2} \quad a_{n,3} \quad \cdots \quad a_{n,n-1} \quad 0\,) + (0 \ldots 0\ 1)]Q_{n-1}^{-1} \\
&= (\, a_{n,2} \quad a_{n,3} \quad \cdots \quad a_{n,n}\,) \times Q_{n-1}^{-1}
\end{aligned}
$$

Tenemos que verificar que

$$(\, a_{n,2} \quad a_{n,3} \quad \cdots \quad a_{n,n}\,) \times Q_{n-1}^{-1} = (\, 1 \quad 1 \quad \cdots \quad 1\,)$$

y esto significa probar

$$(\, a_{n,2} \quad a_{n,3} \quad \cdots \quad a_{n,n}\,) = (\, 1 \quad 1 \quad \cdots \quad 1\,) \times Q_{n-1}$$

esto es

$$(\, a_{n,2} \quad a_{n,3} \quad \cdots \quad a_{n,n}\,) = (\, \textstyle\sum_{i=1}^{n-1} a_{i,1} \quad \cdots \quad \sum_{i=1}^{n-1} a_{i,n-1}\,) \ .$$

Pero esto es cierto, pues sabemos que $\forall\ j \geq 2$

$$a_{n,j} = a_{1,j-1} + \ldots + a_{n-1,j-1}$$

y con esto hemos terminado la comprobación de que $T = Q \times J \times Q^{-1}$.

Sólo nos resta trasladar los resultados a $\mathbf{Z_2}$. Hemos obtenido una matriz Q tal que $T = Q \times J \times Q^{-1}$ en $\mathbf{R^n}$. Podemos aplicar tranquilamente la operación *modulo 2* a las matrices que aparecen a ambos lados de la igualdad y el resultado seguirá siendo cierto, pues $\mathbf{Z_2}$ es un cuerpo con las operaciones suma y producto. Las matrices T y J siguen teniendo la misma forma tras aplicar la operación *modulo 2*, y la matriz Q pasa a verificar $Q = Q^{-1}$, lo que es fácilmente comprobable si recordamos que al calcular Q^{-1} hemos demostrado que llamando $Q = (a_{i,j})_{i,j=1\ldots n}$, $Q^{-1} = (b_{i,j})_{i,j=1\ldots n}$, se tiene que $b_{i,j} = (-1)^{i+j} a_{i,j}$.

Por tanto la expresión de la matriz Q en $\mathbf{Z_2}$ es:

$$Q \quad = Q^{-1} \quad = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & \cdots & \cdots & 0 & 0 & 0 \\ \vdots & & & & & & & & & \vdots \\ \vdots & & & & & & & & & \vdots \\ & & & & & \ldots & \cdots & & 1 & 0 \\ & & & & & \ldots & \cdots & & & 1 \end{pmatrix}$$

Como ejemplo, la matriz Q de dimensión 5 es:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

y la de dimensión 9 :

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

6.2 Reformulación del Problema

Si observamos la matriz Q obtenida en el apartado anterior, podemos apreciar que eliminando la primera fila y columna de la matriz y quedándonos con la mitad inferior de la matriz restante - la mitad superior sólo contiene ceros - obtenemos un fragmento del triángulo de Pascal en $\mathbf{Z_2}$. Lógico si recordamos que al calcular la forma de Q hemos llegado a la conclusión de que el elemento situado en la posición (i,j) es precisamente $\binom{i-2}{j-2}$.

En el caso de la matriz Q de dimensión 5 tendríamos:

$$\left(\begin{array}{ccc|cc} 1 & & & & \\ 1 & 1 & & & \\ - & - & & - & - \\ 1 & 0 & & 1 & \\ 1 & 1 & & 1 & 1 \end{array}\right) = \left(\begin{array}{ccc|cc} \binom{0}{0} & & & & \\ \binom{1}{0} & \binom{1}{1} & & & \\ - & - & & - & - \\ \binom{2}{0} & \binom{2}{1} & & \binom{2}{2} & \\ \binom{3}{0} & \binom{3}{1} & & \binom{3}{2} & \binom{3}{3} \end{array}\right)$$

y en el caso de la matriz de dimensión 9 :

$$\left(\begin{array}{cccc|cccc} 1 & & & & & & & \\ 1 & 1 & & & & & & \\ 1 & 0 & 1 & & & & & \\ 1 & 1 & 1 & 1 & & & & \\ - & - & - & - & - & - & - & - \\ 1 & 0 & 0 & 0 & 1 & & & \\ 1 & 1 & 0 & 0 & 1 & 1 & & \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right) = \left(\begin{array}{cccc|cccc} \binom{0}{0} & & & & & & & \\ \binom{1}{0} & \binom{1}{1} & & & & & & \\ \binom{2}{0} & \binom{2}{1} & \binom{2}{2} & & & & & \\ \binom{3}{0} & \binom{3}{1} & \binom{3}{2} & \binom{3}{3} & & & & \\ - & - & - & - & - & - & - & - \\ \binom{4}{0} & \binom{4}{1} & \binom{4}{2} & \binom{4}{3} & \binom{4}{4} & & & \\ \binom{5}{0} & \binom{5}{1} & \binom{5}{2} & \binom{5}{3} & \binom{5}{4} & \binom{5}{5} & & \\ \binom{6}{0} & \binom{6}{1} & \binom{6}{2} & \binom{6}{3} & \binom{6}{4} & \binom{6}{5} & \binom{6}{6} & \\ \binom{7}{0} & \binom{7}{1} & \binom{7}{2} & \binom{7}{3} & \binom{7}{4} & \binom{7}{5} & \binom{7}{6} & \binom{7}{7} \end{array}\right)$$

Vamos a dar una definición de la matriz Q que saca partido de la recursividad del triángulo de Pascal en $\mathbf{Z_2}$, de la que hemos querido dejar constancia con los trazos que dividen en cajas las matrices de los dos ejemplos anteriores.

Llamamos :

- $P^0 = (1)$
- $P^{k+1} = \begin{pmatrix} P^k & 0^k \\ P^k & P^k \end{pmatrix}$, siendo 0^k la matriz cero de tamaño 2^k.

Definimos la matriz Q de dimensión (2^k+1) - nos referiremos a ella como $Q_{2^{k+1}+1}$ - en función de la recién definida P como:

$$Q_{2^{k+1}+1} = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ - & - & & - \\ 0 & P^k & & 0^k \\ \vdots & & & \\ 0 & P^k & & P^k \end{array}\right)$$

Una vez conocida la aplicación lineal Q, el problema de las sumas parciales en el caso semigrupo se puede plantear como el estudio de parejas de matrices $(Q \times A)$ y $(B \times Q)$ tales que $A \times B = J$.

Capítulo 7

Problemas Abiertos

El problema de las sumas parciales en el caso semigrupo, y dentro del modelo algebraico al que nos hemos referido en este trabajo, está totalmente comprendido. Se conoce incluso la cota inferior optimal para el coste medio asumiendo distintas distribuciones de probabilidad para las operaciones (ver [15]). De hecho estos resultados son válidos para un modelo más general que incluye al modelo algebraico definido por M.L.Fredman: se asume la existencia de variables $z_1, z_2 \ldots$ que almacenan valores en el semigrupo, y que permiten hacer operaciones del tipo $z_i \leftarrow z_j + z_k$.

Hay muchos problemas abiertos si se consideran modelos menos restrictivos. Por ejemplo, si consideramos el caso de grupo, existe una brecha entre la mejor cota superior obtenida y la mejor cota inferior para la complejidad media de las operaciones. En el Capítulo 6 de esta tesis hemos sugerido una posible vía para atacar el problema de manera distinta, aunque hemos de reconocer que no parece más sencilla que otras anteriores. Naturalmente el problema se complica si asignamos distintas probabilidades a la ejecución de las distintas operaciones.

De manera similar, si el problema se trata dentro del *cell probe model of computation* mencionado en el Capítulo 1, y para ciertos parámetros del problema (tamaño de palabra de memoria, tipo de valores almacenados...), existen las mismas distancias entre las mejores cotas superior e inferior.

En cuanto al problema más general de los rangos variables, y dentro del modelo en el que hemos trabajado en el Capítulo 5, permanecen abiertos bastantes

problemas - más aún si consideramos que se almacenan valores pertenecientes a un grupo en lugar de a un semigrupo. Exponemos algunos de ellos a continuación, pues simplemente vamos a enumerar los nuevos problemas que se nos han planteado como consecuencia directa del trabajo que hemos realizado. Algunos de ellos son de carácter teórico, pero otros son eminentemente prácticos.

Como continuación de nuestro trabajo nos proponemos abordar las siguientes cuestiones:

- Aplicar la técnica de análisis de matrices desarrollada en el Capítulo 3 al modelo matricial que hemos definido para el problema de los rangos variables. El objetivo es encontrar una cota inferior para el número de unos en una pareja R, U de matrices de ceros y unos cuyo producto sea la matriz H, y probar su optimalidad si es que es posible. Equivale a encontrar una cota inferior optimal para la complejidad del conjunto de operaciones asumiendo una distribución de probabilidad uniforme.

- Implementación eficiente - generar código - de soluciones distintas definidas en el marco del modelo orbital que hemos definido en el Capítulo 4.

- Implementación eficiente - programas - de las soluciones definidas por los grafos y matrices del Capítulo 5.

- Estudio del problema de los rangos variables introduciendo distintas distribuciones de probabilidad para las operaciones, dentro de los modelos matricial y de grafos que hemos definido en el Capítulo 5.

- Estudiar si los grafos y matrices que hemos definido en el Capítulo 5 son optimales en términos de complejidad media conjunta para las operaciones.

- Intentar abordar el problema de las sumas parciales en el caso grupo partiendo de las ideas sugeridas en el Capítulo 6.

Bibliografía

[1] A. V.Aho, J. E.Hopcroft, J. D.Ullman, *The design and analysis of computer algorithms*, Addison-Wesley, 1974.

[2] A.Andersson, M.Thorup, *Tight(er) worst case bound on dynamic searching and priority queues* , Proc. 32th Annual ACM Symposium on Theory of Computing (STOC´00).

[3] S. Alstrup, A. Ben-Amram, T. Rauhe, *Worst case and amortized optimality in Union-Find*, Proc. STOC´99, 499-506.

[4] G. Brassard, P. Bratley, *Algorithmics: Theory and Practice*, Prentice-Hall, 1988.

[5] A. Brodnik, J. Munro, *Membership in constant time and almost-minimum space*, Siam J. Comput. Vol. 28, 1999, 1627-1640.

[6] W. A. Burkhard, M. L. Fredman, *Inherent complexity trade-offs for range query problems*, Theoretical Computer Science, North Holland Publishing company, 1981, in Vol.16, 279-290.

[7] T. H. Cormen, Ch. E. Leiserson, R. L. Rivest, *Introduction to Algorithms*, The MIT Press, McGraw-Hill Book Company.

[8] P.Dietz, *Optimal algorithms for list indexing and subset ranks*, WADS, 1989.

[9] M. Dietzfelbinger, A. Karlin, K. Mehlhorn, F. Meyer, *Dynamic perfect hashing: upper and lower bounds*, Siam J. Comput. Vol.23, 1944, 738-761.

[10] Y.DODIS, *Space-Time tradeoffs for graph properties*, Master´s Thesis, MIT, 1998.

[11] G. S. FRANDSEN, J. P. HANSEN, P. B.MILTERSEN, *Lower bounds for dynamics algebraic problems*, STACS´99, Lecture Notes in Computer Science, Vol. 1562, 1999, 362-372.

[12] M. L.FREDMAN, *A lower bound on the complexity of orthogonal range queries*, J. ACM Vol.28, No.4, 1981, 696-705.

[13] M. L. FREDMAN, *Lower bounds on the complexity of some optimal data structures*, Siam J. Comput. Vol.10, No.1, 1981, 1-10.

[14] M. L. FREDMAN, *The complexity of maintaining an array and computing its partial sums*, J.ACM, Vol.29, No.1, 1982, 250-260.

[15] M. L. FREDMAN, H. HAMPAPURAM, *Optimal Bi-Weighted Binary Trees and the Complexity of Maintaining Partial Sums*, Siam J. Comput. Vol.28, No.1, 1998, 1-9.

[16] M. L. FREDMAN, D. S. JOHNSON, L. A. MCGEOCH, G. OSTHEIMER, *Data structures for Travelling Salesmen*, Journal of Algorithms, Vol.18, 1995, 432, 479.

[17] M. L. FREDMAN, J.KOMLOS, E.SZEMEREDI, *Storing a sparse table with O(1) worst case access time*, J. ACM, Vol.31, 1984, 538-544

[18] M. L. FREDMAN, M. E. SAKS, *The cell probe complexity of dynamic data structures*, Proceedings of 21 st ACM Symposium on Theory of Computing, 1989, 345-354.

[19] M. L. FREDMAN, D. J. VOLPER, *Query Time Versus Redundancy Tradeoffs for Range Queries*, Journal of Computer and System Sciences, 1981, in Vol.23, 355-365.

[20] Z. GALIL, G. F. ITALIANO , *Data structures and algorithms for disjoint set union problems*, STACS´98, Lecture Notes in Computer Science, Vol. 1372, 366-398.

[21] T. HAGERUP, *Sorting and searching on the word RAM*, STACS´98, Lecture Notes in Computer Science, Vol. 1372, 366-398.

[22] D. E. KNUTH, *Fundamental Algorithms*, Vol.1 of *The Art of Computer Programming*, Addison-Wesley, 1968.

[23] J. LANDIN, *An introduction to algebraic structures*, Dover Publications, Inc., New York.

[24] W. NEF, *Linear Algebra*, McGraw-Hill Publishing Company Limited.

[25] R. PAGH, *Low redundancy in static dictionaries with O(1) lookup time*, ICALP´99, Lecture Notes in Computer Science, Vol. 1644, 1999, 595-604.

[26] J. H. REIF, S. R. TATE , *On dynamic algorithms for algebraic problems*, Journal of Algorithms, Vol. 22, 1997, 347, 371.

[27] G. E. SHILOV, *Linear Algebra*, Dover Publications, Inc., New York.

[28] R. TAMASSIA, F.PREPARATA, *Dynamic maintenance of planar digraphs, with applications*, Algorithmica, Vol. 5, 1990, 509-527.

[29] A. TONI, *Lower bound on zero-one matrices*, to appear in Linear Algebra and its Applications.

[30] A. TONI, *Average complexity of addition updates and range queries over an array using graphs*, Proceedings CTW'03, publicado en Electronic Notes in Discrete Mathematics, Vol. 13, 2003.

[31] A. TONI, *Matricial model for the range query problem* , Proceedings of the International Workshop on Combinatorics, Linear Algebra and Graph Coloring, Institute for Studies in Theoretical Physics and Mathematics, Tehran, 2003.

[32] A. TONI, *Class of programs for update and retrieve operations over an array* , Proceedings del Workshop Argentino en Informática Teórica (WAIT'99), Universidad de La Plata, Argentina 1999.

[33] R.WILBER, *Lower bounds for accessing binary trees with rotations*, SiamJ. Comput. Vol.18, 1989, 56-67.

[34] D. WILLARD, *Log-logarithmic worst case range queries are possible in space* $\Theta(n)$, Inform. Process. Lett. Vol.17, 1983, 81-84.

[35] A. C.YAO, *On the Complexity of Maintaining Partial Sums*, SiamJ. Comput. Vol.14, No.2, 1985.

[36] A. C.YAO, F. F.YAO, *Dictionary look-up with one error*, Journal of Algorithms, Vol.25, 1997, 194-202.

Índice General